Printed & Bound by
Abbey Bookbinding
Unit 3 Clos Menter
Gabalfa Workshops
Western Avenue
Cardiff CF14 3AY
T: +44 (0) 29 2062 3290
F: +44 (0) 29 2062 5420
E: info@bookbindersuk.com
W: www.bookbindersuk.com

Evolution of Loosely Synchronized Spreading Codes in Code-division Multiple-access Systems

Richard Peter Ward
Division of Mathematics and Statistics
University of Glamorgan
Pontypridd, CF37 1DL
Wales, U.K.

A submission presented in partial
fulfilment of the requirements of the
University of Glamorgan/Prifysgol Morgannwg
for the degree of Doctor of Philosophy

This research programme was
carried out in collaboration
with BAE SYSTEMS

July 11, 2008

Acknowledgements

I would like to thank my director of studies Prof. Derek Smith for being extremely generous with his time, for his support and every little thing that contributed to the success of this work. My appreciation also goes to my supervisors Dr. Stephanie Perkins, Mr. Glyn Wyman and to Dr. Richard Jones for all of their time, knowledge, and ideas. I would also like to thank EPSRC for funding my CASE studentship and BAE SYSTEMS for their support and collaboration.

I want to express my gratitude to the Division of Mathematics and Statistics for giving me the opportunity to study here and to find great friends here. My best regards also to the all staff of the Faculty of Advanced Technology and my fellow research colleagues who I have shared the past few years with.

Finally to Aimee who has had to share this thesis day in day out.

Abstract

Loosely Synchronized (LS) codes can be used as spreading codes in quasi-synchronous code-division multiple-access (QS-CDMA) systems. In such CDMA systems, close control of synchronization is achieved at the chip level, intermediate between that in synchronous CDMA and that in asynchronous CDMA. The LS code can then capitalize on zero correlation in a limited synchronization window to reduce code correlations and so reduce interference. LS codes are $\{0, +1, -1\}$ codes constructed using *Hadamard matrices* and *Golay pairs*. A variation of LS codes inserts short strings of zeros between the components of the Golay pairs to increase the number of codewords, with only limited deterioration in the correlations. These strings of zeros are known as *internal padding*.

One of the advantages normally claimed for CDMA systems is resistance to eavesdropping and jamming. It might appear at first sight that the structure of LS codes is rather predictable in comparison with codes constructed using linear feedback shift registers, such as m-sequences or Gold codes. One way to overcome any such difficulty would be to evolve the code very quickly, in such a way that by the time a generation of the code is determined (or determined to a moderate correlation value) it is too late to exploit it. This thesis explores the way that LS codes can be evolved in order to achieve resistance to eavesdropping and jamming.

The thesis starts with a detailed account of the necessary background and of the construction of Loosely Synchronized codes. The early part of the thesis then concentrates on showing that many generations of LS code can be constructed in such a way that the correlation between distinct generations is small. This prevents one observed generation of the code from being used for jamming or prediction in another generation. Specifically:

- The construction of Golay pairs is investigated and a search is carried out over all possible Golay pairs and their mates to find a set of pairs that leads to the satisfaction of a suitable correlation criterion;
- Bent functions, almost bent functions and other second order Boolean functions are used to create sets of Hadamard matrices that are guaranteed to

satisfy the same correlation criterion;

- A sequential search method to generate a set of arrangements of the internal padding that satisfies the same correlation criterion is described. Later in the thesis this approach is replaced by a *recency list* approach. This ensures that the correlation criterion is satisfied against recently used generations of the code, in place of all generations of the code;

- The way in which these evolutions of the components combine together is also explored.

Attention turns in the second part of the thesis to the mechanisms for evolution and the way that these might be predicted by a third party observer. Transform methods that the third party might use are described. Detailed simulations quantify the ability of the third party to identify the code during the transmission of a single bit. It is shown that theoretical resistance to early code prediction is not possible, although it might be possible to demonstrate security arising from the relative speed of the necessary computations for the user and the observer. This would require a detailed hardware study, and this is listed as future work. In fact it is shown here that LS codes are actually better than linear feedback shift register codes, as a result of the Berlekamp-Massey algorithm.

Attention is also focussed on the scenario in which details of the algorithms of one user are obtained by the third party. Only the Hadamard matrix provides protection against this scenario, as all other components of the construction are shared between all users. From this second viewpoint the true weakness of LS codes becomes apparent. Although the Hadamard matrix constructions are satisfactory if the order of the Hadamard matrix is not too small, it seems that the sequence of Hadamard matrix rows of each user must be computed centrally and distributed to users as private keys if this scenario is not to remain a major concern. The volume of private key distribution necessary may seem unattractive to operators.

Ultimately it seems that evolution of the Golay pairs may have little real role except to increase the workload of the observer. The recency list based evolution of internal padding can take the main role in ensuring low correlation between close generations of the code. The evolution of the Hadamard matrix should be designed to concentrate on the second viewpoint, where the third party has obtained details of the algorithms of one user.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Coding theory has many important applications in multiple-access radio systems. These applications are not restricted to error-control coding. This thesis addresses an application concerned with the security of certain code-division multiple-access (CDMA) systems.

1.1 Multiple-access radio systems

Multiple-access defines the method by which more than one user is allowed to share the available resources in a radio system. Four major multiple-access techniques are available in the literature [19]:

- frequency-division multiple-access (FDMA);
- time-division multiple-access (TDMA);
- frequency hopping code-division multiple-access (FH-CDMA);
- direct sequence code-division multiple-access (DS-CDMA).

In FDMA, each user's signal is transmitted over a single narrow band of frequencies which is sufficiently separated from adjacent users' frequencies for interference to be reduced to tolerable levels. A frequency can be re-used when two users are a sufficient distance apart.

In TDMA, signals are transmitted over the entire available frequency bandwidth at different times. This time difference prevents interference.

In FH-CDMA, a transmitter "hops" between available frequencies according to a specified algorithm, which can be either pseudo-random or preplanned. In this case a non-binary code is used to define the hopping sequences. The transmitter operates in synchronization with a receiver, which remains tuned

to the same centre frequency as the transmitter. Data is modulated on a carrier in a time slot. This carrier is then changed in the next time slot in a way that is consistent with the hopping sequence. Energy is observed over a wide spectral range. The transmitter is capable of hopping its frequency over a given bandwidth several or very many times a second, transmitting on one frequency for a certain period of time, then hopping to another frequency and transmitting again. Frequency hopping requires a much wider bandwidth than is needed to transmit the same information using only one carrier frequency. The hopping algorithm is designed to minimize interference between users. The main advantage of frequency hopping is frequency diversity, which allows error-correction to perform more effectively. Frequency hopping can also have some limited security advantages.

In DS-CDMA, all signals are transmitted and spread over the same available wide band of frequencies at the same time. To differentiate signals, each transmitter is assigned a spreading codeword (spreading sequence) which is different from the spreading codewords assigned to other transmitters within an allowable re-use distance. An intended receiver is able to differentiate its wanted spread spectrum signal from unwanted spread spectrum signals as a result of the favourable correlations between the spreading codewords used (explained in more detail in section 1.2). In principle a receiver is able to demodulate the transmitted sequence by using a correlator at the receiver, which has the advantage of spreading the energy of signals with different codes.

Hybrids of the above systems also exist. In this thesis the multiple-access system technique studied is DS-CDMA, which will simply be referred to as CDMA.

First generation (1G) wireless communication systems use analogue methods. These systems superimpose the message signal onto the radio frequency (RF) carrier using frequency modulation and separate users by FDMA techniques. An example of this type of system is the Advanced Mobile Phone System (AMPS). AMPS was the first system to implement a cellular structure and to assign voice channels adaptively. The use of paired channels was introduced.

In September 1988, the Cellular Telecommunications Industry Association (CTIA) laid out user performance requirements for digital systems, including:

- Tenfold increase in capacity over analogue systems;
- Ability to introduce new features;
- Higher voice quality;

- Voice and data protection;
- Ease of transitions and compatibility with existing analogue systems.

Second generation (2G) communication systems introduce digital technology. These systems digitally encode the message signal before superimposing it onto the RF carrier. Digital data allows coding techniques that both improve voice quality and increase network capacity. Examples of this type of system include GSM (Global System for Mobile Communications) and CDMA IS-95.

Third generation (3G) communication systems introduce soft factors and packet-switched connections that are dynamic and more efficient. A packet-switching network assigns resources based on the data throughput demands and the quality of service requirements of each application. The network efficiently maps the available resources to the needs of each application using improvements in several key areas, including frequency diversity, flexible data rates, spreading techniques, source or error correction coding, and reverse-link coherent detection.

Developments beyond 3G such as Super 3G and Pre 4G point the way to 4G systems, which will be entirely internet protocol (IP) based. These developments aim to offer high-speed internet access that rivals fixed-line communications and may be critical in the success of mobile television and video.

CDMA predates 3G systems. In fact satellite systems used CDMA in the 1970s. However, CDMA has achieved particular commercial importance as a result of the introduction of third generation systems. CDMA is a wideband spread spectrum system. As such, it spreads the radio signal over a frequency bandwidth that is much wider than is necessary to transfer the information sent. The wave form used in a CDMA system can be pseudorandom. CDMA codes are designed to have very low cross-correlation. Some features of CDMA are: capacity advantage, graceful degradation, multipath resistance, inherent frequency diversity, security against eavesdropping and jamming, interference rejection, and the potential use of advanced antenna and receiver structures.

1.2 Spreading codewords and definitions of correlation

Definition 1 *A codeword is a vector* $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ *with* $x_i \in \{0, 1\}$, $x_i \in \{-1, +1\}$ *or* $x_i \in \{0, -1, +1\}$.

Definition 2 *A code is a set of codewords.*

A codeword sometimes arises as a single period of an infinite periodic sequence.

Spreading codewords are used in a CDMA system for spreading the signal at the transmitter and for despreading the signal at the receiver. This spreading codeword is usually a $\{+1, -1\}$ binary codeword, although in this thesis $\{0, +1, -1\}$ ternary codes with a limited number of 0's will be used. Each information bit is replaced by the spreading codeword or the negative of the spreading codeword. For example, if two information bits $+1, -1$ are to be transmitted in a CDMA system and the spreading code is $+1, +1, -1, +1, -1, +1, +1, -1, +1, -1, -1$ then the binary signal (before any modulation) is shown in figure 1.1. The two information bits are replaced by 22

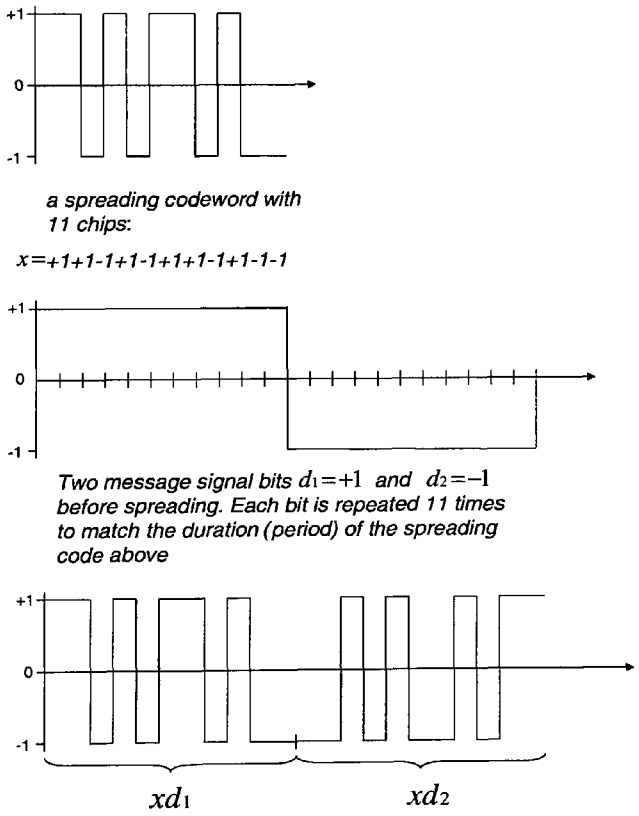

Figure 1.1: *An illustration of a spreading codeword, two message bits and the spread spectrum signal formed.*

$\{+1, -1\}$ values referred to as *chips*, in the same time interval. The fact that the chip rate is then much higher than the bit rate effects the spreading of the signal.

An intended receiver is able to differentiate its wanted spread spectrum signal (with an information bit $+1$ replaced by $\mathbf{x}$) from unwanted signals (with information bits $+1$ replaced by $\mathbf{y}$) as a result of the correlations between the spreading codewords used. Assume perfect synchronization and that $\mathbf{x}$ and $\mathbf{y}$ are orthogonal (correlation zero). The receiver of the wanted signal decodes the bit by computing $(\mathbf{x}.\mathbf{x})/n = +1$. If synchronization is not perfect it is necessary to lock on to the codewords, making use of the peaked nature of the autocorrelation function. Low off-peak autocorrelation values are necessary to mitigate any multi-path interference (between a signal and a delayed version of the same signal). Unwanted signals are rejected as a result of low cross-correlation values (giving low multiple-access interference). Thus a receiver using spreading codeword $\mathbf{y}$ would compute $(\mathbf{x}.\mathbf{y})/n = 0$ and reject the first signal. Zero off-peak autocorrelation values and zero cross-correlation values are ideal, but values close to zero can be tolerated. A limiting factor when using spreading codes where the cross-correlations are not zero is that interference increases with the number of users. This can severely limit the capacity of CDMA systems.

Interference in CDMA thus depends on the correlation properties of the spreading code used. Two commonly used families of spreading codes are *m-sequences* and *Gold codes.*

Definition 3 *An m-sequence is a (nonzero)* $\{0, 1\}$ *periodic binary sequence generated by a binary linear recurrence relation (or a binary linear feedback shift register) with a primitive connection polynomial. Any single period of the sequence forms a codeword of the spreading code.*

Definition 4 *A Gold code is a set of* $\{0, 1\}$ *periodic sequences obtained from an m-sequence and a* $2^r + 1$ *decimation of the m-sequence. The decimation is obtained by taking every* $2^r + 1$*th position of the m-sequence. The Gold code is obtained by taking a sum of cyclic shifts of the m-sequence and of its decimation in all possible ways.*

Further information on spreading codes in CDMA systems can be found in [7], [18].

Definition 5 *The signal-to-noise ratio for a receiver is the ratio of the power of the wanted signal at the receiver (after decoding) to the sum of the powers of all noise signals. In a CDMA system the noise is often considered as the sum of the background noise and all interfering signals at the receiver. The signal-to-noise ratio is usually expressed in decibels.*

Good correlation properties of the spreading code lead to a high signal-to-noise ratio. High signal-to-noise ratio leads to a low bit error rate.

There are three different ways to measure the correlation of codewords. These are the even, odd and aperiodic correlations and each can be used to calculate the auto or cross-correlation of particular codewords.

Definition 6 *Define* $\theta_{x,x}(\tau)$,$\hat{\theta}_{x,x}(\tau)$ *and* $C_{x,x}(\tau)$ *as the even, odd and aperiodic autocorrelations between a spreading codeword* x *of length* n *and a shift of itself by* τ *chips. Here* $x_{i+\tau}$ *will generally be taken to denote* $x_{i+\tau \bmod n}$.

$$\theta_{x,x}(\tau) = \sum_{i=0}^{n-1} x_i x_{i+\tau} : (1-n \leq \tau \leq n-1).$$

$$\hat{\theta}_{x,x}(\tau) = \sum_{i=0}^{n-\tau-1} x_i x_{i+\tau} - \sum_{i=n-\tau}^{n-1} x_i x_{i+\tau} : (0 \leq \tau \leq n-1).$$

$$C_{x,x}(\tau) = \begin{cases} \sum_{i=0}^{n-1-\tau} x_i x_{i+\tau} & : \quad 0 \leq \tau \leq n-1 \\ \sum_{i=0}^{n-1+\tau} x_{i-\tau} x_i & : \quad 1-n \leq \tau < 0 \\ 0 & : \quad \mid \tau \mid \geq n. \end{cases}$$

Define $\theta_{x,y}(\tau)$,$\hat{\theta}_{x,y}(\tau)$ *and* $C_{x,y}(\tau)$ *as the even, odd and aperiodic cross-correlations between spreading codewords* x *and* y *of length* n.

$$\theta_{x,y}(\tau) = \sum_{i=0}^{n-1} x_i y_{i+\tau} : (1-n \leq \tau \leq n-1).$$

$$\hat{\theta}_{x,y}(\tau) = \sum_{i=0}^{n-\tau-1} x_i y_{i+\tau} - \sum_{i=n-\tau}^{n-1} x_i y_{i+\tau} : (0 \leq \tau \leq n-1).$$

$$C_{x,y}(\tau) = \begin{cases} \sum_{i=0}^{n-1-\tau} x_i y_{i+\tau} & : \quad 0 \leq \tau \leq n-1 \\ \sum_{i=0}^{n-1+\tau} x_{i-\tau} y_i & : \quad 1-n \leq \tau < 0 \\ 0 & : \quad \mid \tau \mid \geq n. \end{cases}$$

In this thesis it will be convenient to work with *normalized* aperiodic correlations.

Definition 7 *Define*

$$\theta^{(a)}_{x,y}(\tau) = \frac{C_{x,y}(\tau)}{C_{x,x}(0)}$$

as the normalized aperiodic cross-correlation (scaled between 0 and 1). Similarly $\theta^{(a)}_{x,x}(\tau)$ *is the normalized aperiodic autocorrelation.*

Definition 8 *Let*

$$C_{x,y} = \max_{\tau} |C_{x,y}(\tau)|, -T < \tau < T, x \neq y.$$

Then

$$\theta_{x,y}^{(a)} = \frac{C_{x,y}}{C_{x,x}(0)}$$

denotes the maximum (or peak) normalized aperiodic cross-correlation (in an interval (−T,T) for τ). Also $\theta_{x,x}^{(a)} = \theta_{x,x}^{(a)}(0)$ is the maximum (or peak) normalized aperiodic autocorrelation.

Autocorrelation refers to the degree of correspondence between a sequence x and a shifted replica of itself. Cross-correlation measures the agreement or disagreement between two different sequences x and y (where y is shifted by τ positions).

For example, if:

$$\mathbf{x} = (+1, -1, -1, -1, +1, +1, +1)$$
$$\mathbf{y} = (+1, +1, -1, +1, -1, +1, +1)$$

then

$$C_{x,x}(3) = (+1. - 1) + (-1. + 1) + (-1. + 1) + (-1. + 1) = -4$$
$$C_{x,y}(3) = (+1. + 1) + (-1. - 1) + (-1. + 1) + (-1. + 1) = 0.$$

In the binary case the correlation measures are often thought of as counting "agreements–disagreements". This needs modification in the case of $\{0, +1, -1\}$ ternary codes. "Agreements–disagreements" are only counted in positions in which neither codeword is zero. Aperiodic cross-correlation is (for $\tau > 0$) the mean of the even and odd correlation. It should ideally be used as a measure of interference in a CDMA system [18].

1.3 Jamming and eavesdropping

Apart from their advantages for multiple-access, two of the advantages usually quoted for CDMA systems are their resistance to jamming and eavesdropping [19].

Jamming is the transmission of signals by a third party that disrupt communications by decreasing the signal-to-noise ratio to unsatisfactory levels. In a CDMA system in which the jammer has no knowledge of the spreading codeword, the jamming signal is spread over the entire spectrum. However, the power of the interfering signal is scaled down on decoding in proportion to the length of the spreading codeword. Thus jamming in a CDMA system is very

ineffective if it is possible to ensure that little useful knowledge of the spreading codeword is available to the jammer.

Eavesdropping is the interception of a message by unintended recipients in order to determine the information contained therein. In a CDMA system knowledge of the spreading codeword is necessary to convert the spread spectrum signal to the actual encoded signal. If little knowledge of the spreading codeword is available it is not possible to decode the signal. Thus CDMA systems can provide additional protection against eavesdropping to any cryptographic protection present. Again to achieve this additional protection, it must be ensured that little useful knowledge of the spreading codeword is available to the eavesdropper.

1.4 Synchronization

In synchronous CDMA (S-CDMA) system time is defined and synchronization can be maintained at much less than the chip level (i.e. much less than the level of the individual elements of the codewords). In terrestrial systems this level of synchronization cannot be maintained as a result of differences in propagation delay, and asynchronous CDMA (A-CDMA) is used. The codes used inevitably have worse correlation properties and as a consequence the level of interference from multiple sources in the network may become undesirably high. An intermediate option referred to as quasi-synchronous CDMA (QS-CDMA) [6, 21, 25] ensures that the synchronization uncertainty is constrained to a small number of chips. This will be described in the next section.

1.5 Quasi-synchronous CDMA and Loosely Synchronized codes

In QS-CDMA, the relative time delays of interfering signals are confined to small values. There may be a knowledge of a global clock, such as a GPS clock, so that when a chip i is sent the actual chip received is within the uncertainty range of $\{i - \tau_{max} + 1, \ldots, i + \tau_{max} - 1\}$, where τ_{max} is a small integer. Control over synchronization uncertainty may also be effected by signalling between neighbouring cells, but the ability to synchronize accurately may be limited by the speed of light in systems covering a large area. Spreading codes for quasi-synchronous systems are designed to have zero correlation (or low correlation) within the uncertainty range. The window for zero correlation (or low correlation) is known as a zero (low) correlation zone. The main limiting factor on spreading codes with a zero (low) correlation zone is that the number of codewords available is bounded by a variation of the Welch bound [27]. The bound given in [23] shows

that for a zero correlation zone to exist $(u-1)\tau_{max} \leq n-1$ where u is the number of codewords, n is the length of the codeword and the synchronization uncertainty τ satisfies $|\tau| < \tau_{max}$. Thus the number of codewords is inherently limited. An example of a code with a zero correlation zone will be seen later in figure 1.3.

The possibility of operating a CDMA system quasi-synchronously has led to the construction of codes with a low correlation window. The most attractive of these have a zero correlation window for the aperiodic correlation and are ternary {0,+1,-1} codes. They are called *Loosely Synchronized* (LS) codes [21]. They have a zero correlation zone which allows them to exploit the limited synchronization uncertainty. Specifically this means that a value $\tau_{\max}$ can be defined such that the LS codes have zero correlations for $-\tau_{\max} < \tau < \tau_{\max}$.

LS codes of necessity have a number of zeros inserted known as *external padding*. The inherently small number of codewords can be mitigated by the insertion of additional zeros known as *internal padding*. This doubles the number of codewords with some loss of the ideal correlation properties within the zone and a small increase in length [17]. The modified LS codes will be described in detail in chapter 2. These modified codes will be used in this investigation and will still simply be referred to as Loosely Synchronized codes. The two sets of zeros taken together are simply referred to as *padding*.

The constructions of such codes are based on structures known as Golay pairs and Hadamard matrices. One disadvantage of LS codes is that they appear to be rather predictable, and thus provide less protection against jamming and eavesdropping. One solution is to allow the codes to evolve by varying the code construction continuously. The period of the evolution should be as large as possible to avoid the observer exploiting any periodicity. The variation can be effected by changing the Golay pair, changing the Hadamard matrix or changing the internal padding in the code. In each case the change must cause a significant difference in each codeword. The cross-correlation between distinct codewords should be close to zero unless the codewords are very well separated in the evolution.

1.6 Aims and scope of the work

Security against jamming and eavesdropping is the main concern of this thesis. The idea of evolving the code in use will be assessed, specifically for the case of LS codes. For full protection this must take place in such a way that a third party cannot determine any codeword in use before it is too late to exploit it. It is also necessary that no codeword in any generation determined by the third

party can be used in place of an unknown codeword in a later generation to jam or eavesdrop. This needs to be achieved whilst maintaining multiple-access capability and low correlation between successive codes.

It has been suggested above that LS codes are more predictable than codes in use today, such as m-sequences or Gold codes. It will be seen later in the thesis that this is not necessarily the case.

A study has been completed of the number of different options for each of the three components which make up the construction of modified LS codes. Initially the total number of possibilities for each component can be determined. However it is more useful to take the correlations into account. Then the set of options for the components must meet the necessary correlation criterion. This criterion is that the correlation between two distinct generations of the code is bounded above. Having determined the options, it is then necessary to develop methods for evolving the code. Such methods will be presented later in this thesis.

1.7 Evolution and correlation between generations

It is important to understand what is meant by the evolution of an LS code. Figure 1.2 presents a simple scheme for evolving a code over a certain length of time. Initially a key exchange occurs and the system is started. All users need to know when a key exchange occurs (the interval of time between key exchanges is labelled as an *epoch* in figure 1.2). After a shorter length of time, possibly as short as the duration of a single information bit, each codeword will change. This change is referred to as an *evolution step* in figure 1.2; evolution is not a change in key but one in which all codewords change to other codewords. This evolution can be as simple as interchanging the codewords assigned to users or as complex as changing all components of the LS code. In the latter case the evolution mechanism causes the entire code to change after every evolution step. It is important that the "change code" step occurs at the same time for all users, so that a single LS code is in use at any one time.

A correlation condition is imposed on distinct pairs of generations of the code. Specifically, consider the codewords assigned to a single user in two distinct generations. For these codewords the aperiodic correlations must not exceed a given threshold. This means that a codeword used in one generation cannot be used to jam transmission in a later generation.

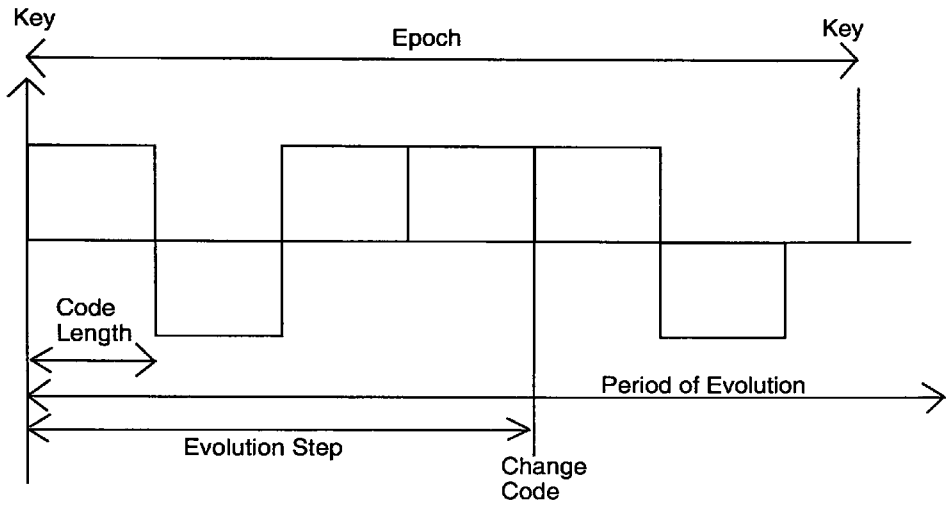

Figure 1.2: *A figure showing the various terms that will be used throughout this thesis.*

Assume that the evolution of the code has period P. Let χ represent the acceptable maximum normalized aperiodic cross-correlation between generations of codewords. This notation will be used throughout this thesis. For any user u assigned codewords $x_u(g_i)$ and $x_u(g_j)$ in generations g_i and g_j it will be required that

$$| \theta^{(a)}_{x_u(g_i)x_u(g_j)} | \leq \chi$$

whenever $i \not\equiv j \bmod P$. Practitioners may choose to use values of χ between 0.1 and 0.5 as a guarantee of adequate levels of security. The actual choice is somewhat arbitrary. In this thesis a "standard" value of 0.25 will normally be used. This should be adequate to provide security in most circumstances and turns out to be a particularly convenient choice.

Within a generation all the codewords have the LS low correlation properties of LS codes with internal padding (see figure 1.3 which presents both the maximum and root mean square correlations). Thus multiple-access interference is very low. This is one of the reasons why LS codes have been chosen for this investigation.

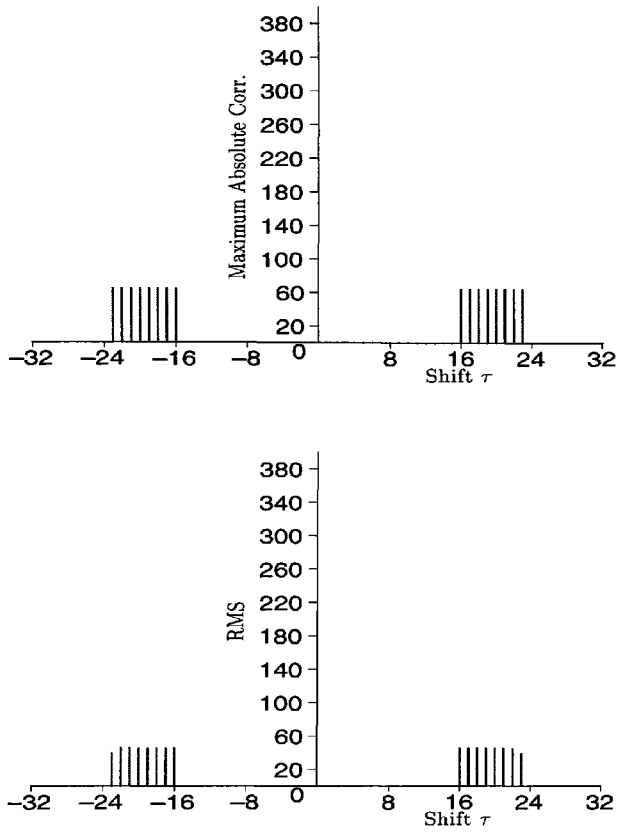

Figure 1.3: *This figure shows (i) the maximum absolute value of the aperiodic cross-correlation against τ in a window $-32 < \tau < 32$, (ii) the root mean square aperiodic cross-correlation against τ in the same window, for pairs of LS codewords in an LS code of length 1296 with internal padding. Such a code is used for a single generation of the evolution.*

1.8 Structure of the thesis

Chapter 2 introduces Hadamard matrices and gives a detailed description of LS codes and a modified construction with internal padding. In chapter 3 results are given for random permutations of the components of the LS code, and Latin squares are introduced. In chapter 4 the possible number of Golay pairs of length a power of 2 is determined. Two constructions are given: one using six operations and another that creates all pairs. Results of clique searches with a suitable correlation criterion satisfied are presented. In chapter 5 it is shown how bent functions and almost bent functions are used to create sets of Hadamard matrices that have low pairwise correlation. Chapter 5 also explains the circumstances

under which a third party might be able to find out the Hadamard matrix in use from knowledge of just one of the Hadamard matrix rows. In chapter 6, sequential searches of the permutations of the internal padding lengths are presented. In chapter 7 a theoretical bound for the correlation between generations of codes in terms of the bounds for the components individually is developed. In chapter 8 various mechanisms and algorithms are given for the evolution of the LS code. In chapter 9 it is shown how a Hadamard transform method can be used to determine all of the components of the current LS codeword. In chapter 10 different mechanisms for evolution of the LS code are investigated, bringing together all components and mechanisms to assess their effectiveness against a third party. Comparisons with m-sequences and Gold codes are also included to show their relative merits. Chapter 11 expands the number of examples considered. Finally, a detailed conclusion and proposals for future work are included in chapter 12.

Chapter 2

Construction of Loosely Synchronized codes

2.1 Components of the construction of Loosely Synchronized codes

A Loosely Synchronized code is a ternary spreading code that can be used in quasi-synchronous CDMA. As stated in section 1.5, LS codes are constructed using Hadamard matrices, Golay pairs and padding.

2.1.1 Hadamard matrices

A *Hadamard matrix* H is an $n \times n$ square matrix with entries ± 1 such that $HH^T = nI_n$ where I_n is the $n \times n$ identity matrix [11, 4]. The rows of the Hadamard matrix form a set of orthogonal binary vectors, which make them ideal for code construction. Since $H^{-1} = (1/n)H^T$ it is also true that $H^T H = nI_n$ and so the columns are also orthogonal. This implies that n must be even, but a more restrictive condition holds: if H is an $n \times n$ Hadamard matrix, then $n = 1$, $n = 2$ or $n \equiv 0 \bmod 4$ [14]. It is conjectured that there is a Hadamard matrix of every order divisible by 4 [4]. In 2005 a Hadamard matrix of order 428 was constructed [12]. This leaves the smallest order n for which no Hadamard matrix has been constructed at 668.

Construction of Hadamard matrices

Denote an $n \times n$ Hadamard matrix by H_n. A simple method used to construct Hadamard matrices H_n, where $n = 2^a$ and a is any positive integer, is the *Sylvester construction* [22]. The current Hadamard matrix is used to produce another matrix of double the size:

$$H_1 = 1$$

$$H_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$H_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

$$H_{2n} = \begin{pmatrix} H_n & H_n \\ H_n & -H_n \end{pmatrix}$$

Note that if H is any Hadamard matrix then the matrix

$$\begin{pmatrix} H & H \\ H & -H \end{pmatrix}$$

is also a Hadamard matrix.

It is also possible to create Hadamard matrices of different sizes using other constructions. For example, the *Paley* [16] and *Williamson* [28] constructions could also be used. The list of constructions given here by no means exhausts those known, but shows that other methods are available if required.

For a Hadamard matrix H_n, either H_n or $-H_n$ must have $n(n-1)/2$ (-1's) and $n(n+1)/2$ ($+1$'s). A negation of a row or column of a matrix is obtained by multiplying it by -1. Two Hadamard matrices are *equivalent* if one can be transformed into the other by a series of row or column permutations and negations. A Hadamard matrix is normalized if all entries in its first row and column are equal to 1.

Theorem 1 *[4] There is a unique equivalence class of Hadamard matrices of each order* $1, 2, 4, 8, 12$. *The number of classes for size* $16, 20, 24, 28, 32$ *and* 36 *are* $5, 3, 60, 487, \geq 66000$ *and* ≥ 110, *respectively.*

The number of equivalent Hadamard matrices in a particular class can be found using the *automorphism group*. This contains all permutations of rows and columns that leave the matrix unchanged when normalized. Choose a starting Hadamard matrix and create the Hadamard matrices obtained from all permutations of the rows and columns. Then normalize each new Hadamard matrix and if the new normalized Hadamard matrix is equal to the original then the size

of the automorphism group is increased by 1. This process is continued until all row and column permutations have been completed. The number of equivalent $n \times n$ Hadamard matrices in the class is then $\frac{n! \times n!}{\nu}$ where ν is the size of the automorphism group. The results are shown in table 2.1. These results show that there are plenty of Hadamard matrices available to use in evolving the code.

n	ζ	ν
4	96	6
8	10752	151200

Table 2.1: *The enumeration of ζ, the number of (equivalent) Hadamard matrices of size $n \times n$, and the size of the corresponding automorphism group.*

2.1.2 Golay pairs

Binary complementary sequences were first introduced by M. J. E. Golay in 1949 [10] (see also [9]) and are defined as pairs of sequences that have aperiodic off-peak autocorrelations that sum to zero. This property makes them ideal in code construction and in many other applications. The term *Golay pair* is used for these binary complementary sequences in this work.

Definition 9 *Let* $\mathbf{a} = (a_0, a_1, ..., a_{N-1})$ *and* $\mathbf{b} = (b_0, b_1, ..., b_{N-1})$ *be a pair of binary sequences of length* N. *The two sequences* $\mathbf{a}$ *and* $\mathbf{b}$ *are complementary (or form a Golay pair) if*

$$C_{\mathbf{a},\mathbf{a}}(\tau) + C_{\mathbf{b},\mathbf{b}}(\tau) = 0 \quad : \quad (1 \leq \tau \leq N).$$

Consider the two sequences of length 2: $\mathbf{a} = (+1, +1)$, $\mathbf{b} = (+1, -1)$

$$\begin{aligned} C_{\mathbf{a},\mathbf{a}}(0) &= +2 & \qquad C_{\mathbf{a},\mathbf{a}}(1) &= +1 \\ C_{\mathbf{b},\mathbf{b}}(0) &= +2 & \qquad C_{\mathbf{b},\mathbf{b}}(1) &= -1 \\ C_{\mathbf{a},\mathbf{a}}(0) + C_{\mathbf{b},\mathbf{b}}(0) &= +4 & \qquad C_{\mathbf{a},\mathbf{a}}(1) + C_{\mathbf{b},\mathbf{b}}(1) &= 0 \end{aligned}$$

Now consider the two sequences of length 4:

$$\mathbf{a} = (+1, +1, +1, -1), \ \mathbf{b} = (+1, +1, -1, +1)$$

$$\begin{aligned} C_{\mathbf{a},\mathbf{a}}(0) &= +4 & \qquad C_{\mathbf{b},\mathbf{b}}(0) &= +4 \\ C_{\mathbf{a},\mathbf{a}}(1) &= +1 & \qquad C_{\mathbf{b},\mathbf{b}}(1) &= -1 \\ C_{\mathbf{a},\mathbf{a}}(2) &= 0 & \qquad C_{\mathbf{b},\mathbf{b}}(2) &= 0 \\ C_{\mathbf{a},\mathbf{a}}(3) &= -1 & \qquad C_{\mathbf{b},\mathbf{b}}(3) &= +1 \\ C_{\mathbf{a},\mathbf{a}}(0) + C_{\mathbf{b},\mathbf{b}}(0) &= +8 & \qquad C_{\mathbf{a},\mathbf{a}}(1) + C_{\mathbf{b},\mathbf{b}}(1) &= 0 \\ C_{\mathbf{a},\mathbf{a}}(2) + C_{\mathbf{b},\mathbf{b}}(2) &= 0 & \qquad C_{\mathbf{a},\mathbf{a}}(3) + C_{\mathbf{b},\mathbf{b}}(3) &= 0 \end{aligned}$$

M. J. E. Golay in 1961 [9] also showed that there are Golay pairs of all lengths

$$2^{\alpha}10^{\beta}26^{\gamma}, \quad \alpha, \beta, \gamma \geq 0.$$

Constructions of sequences whose length is a multiple of 10 and 16 can be found in [8]. Note that the enumeration of Golay pairs for all possible lengths ($N < 100$) has been completed [2]; the results are shown in table 2.2. Detailed constructions of Golay pairs of length 2^{α} will be presented in chapter 4.

N	1	2	4	8	10	16	20	26	32	40	52	64	80
$\mathcal{M}$	4	8	32	192	128	1536	1088	64	15360	9728	512	184320	102912

Table 2.2: *The number $\mathcal{M}$ of Golay pairs of length N.*

2.2 LS codes

The family of Large Area Synchronized (LAS) spreading codes [13] is a combination of Large Area (LA) codes [13] defining pulse positions and Loosely Synchronized (LS) codes [21]. LS codes can be used without combining them with LA codes and only LS codes will be considered in this thesis.

A construction of LS codes by Stańczak, Boche and Haardt [21] will be presented in this section. Let $d = \tau_{max} - 1$ denote the maximum synchronization uncertainty (measured in chips) within the system. The value of d determines the minimum number of zeros in the external padding. The construction creates a ternary $\{0, -1, +1\}$ code. LS codes satisfy $C_{x,y}(\tau) = 0$ for $(0 \leq \mid \tau \mid \leq \tau_{\max} - 1)$ and $C_{x,x}(\tau) = 0$ for $(0 < \mid \tau \mid \leq \tau_{\max} - 1)$. It is easy to achieve this correlation property if sufficient numbers of zeros are used. However for efficient implementation spreading codes must have a *duty ratio* (defined as the number of nonzero elements in a codeword divided by n) close to 1. This arises because transmission of the symbol 0 corresponds to zero information transmission.

An effective way to describe the construction of LS codes is by using generating functions. For example, the sequence $(+1, +1, +1, -1, +1, +1, -1, +1)$ is represented by the generating function $G(z) = 1 + z + z^2 - z^3 + z^4 + z^5 - z^6 + z^7$.

Consider two binary sequences of length N represented by $C(z)$ and $S(z)$.
If

$$C(z)C(z^{-1}) + S(z)S(z^{-1}) = 2N \tag{2.1}$$

(in the Laurent ring $\mathbf{Z}[z, z^{-1}]$ generated by the complex expressions z and z^{-1}), then $C(z)$ and $S(z)$ give a generating function representation of a Golay pair of

length N as defined in the previous section. The two sequences can be treated separately or can be separated with d zeros between them to form a single ternary sequence. These d zeros are referred to as external padding.
For example, consider two sequences of length 4 with generating functions $C(z) = 1 + z + z^2 - z^3$ and $S(z) = 1 + z - z^2 + z^3$. It can be checked that $C(z)C(z^{-1}) + S(z)S(z^{-1}) = 8$ and so $(C(z), S(z))$ is a Golay pair. Then the sequence $+1 + 1 + 1 - 1 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; +1 + 1 - 1 + 1$ with generating function $C(z) + z^{N+d}S(z) = C(z) + z^8 S(z)$ is an LS codeword. Note that it is necessary to add 4 further zeros at the end to generate a periodic codeword (creating further external padding).

A Golay pair of length 2 is given by $A(z) = 1 + z$, $B(z) = 1 - z$. Sequences with length a power of 2 can be obtained using the fact that if $(A(z), B(z))$ represent a Golay pair of length N then a Golay pair $(C(z), D(z))$ of length $2N$ can be created as:

$$C(z) = A(z) + z^N B(z), \quad D(z) = A(z) - z^N B(z). \tag{2.2}$$

For example, if:

$$\begin{aligned} A(z) &= 1 + z + z^2 - z^3 \\ B(z) &= 1 + z - z^2 + z^3 \end{aligned}$$

then a Golay pair of length 8 represented by $(C(z), D(z))$ is obtained:

$$\begin{aligned} C(z) &= A(z) + z^4 B(z) = 1 + z + z^2 - z^3 + z^4 + z^5 - z^6 + z^7 \\ D(z) &= A(z) - z^4 B(z) = 1 + z + z^2 - z^3 - z^4 - z^5 + z^6 - z^7. \end{aligned}$$

Suppose that we have two Golay pairs, represented by $(C_0(z), S_0(z))$ and $(C_1(z), S_1(z))$ such that:

$$\begin{aligned} C_0(z)C_0(z^{-1}) + S_0(z)S_0(z^{-1}) &= 2N, \\ C_1(z)C_1(z^{-1}) + S_1(z)S_1(z^{-1}) &= 2N, \end{aligned} \tag{2.3}$$

and

$$C_0(z)C_1(z^{-1}) + S_0(z)S_1(z^{-1}) = 0. \tag{2.4}$$

The two Golay pairs $(C_0(z), S_0(z))$ and $(C_1(z), S_1(z))$ for which equations 2.3 and 2.4 hold are referred to as *cross complementary sequence pairs* (or simply a pair of complementary sequences) [26], and the second pair is said to be a *mate* of the first. Given a Golay pair $(C_0(z), S_0(z))$, the mate $(C_1(z), S_1(z))$ can be constructed as:

$$C_1(z) = z^{(N-1)}S_0(z^{-1}), \quad S_1(z) = -z^{(N-1)}C_0(z^{-1}). \tag{2.5}$$

For example, if:

$$C_0(z) = 1 + z + z^2 - z^3 + z^4 + z^5 - z^6 + z^7$$
$$S_0(z) = 1 + z + z^2 - z^3 - z^4 - z^5 + z^6 - z^7$$

the mate $(C_1(z), S_1(z))$ is:

$$C_1(z) = z^7 S_0(z^{-1}) = -1 + z - z^2 - z^3 - z^4 + z^5 + z^6 + z^7$$
$$S_1(z) = -z^7 C_0(z^{-1}) = -1 + z - z^2 - z^3 + z^4 - z^5 - z^6 - z^7.$$

Then the Golay pair $(C_0(z), S_0(z))$ with mate $(C_1(z), S_1(z))$ form a cross complementary sequence pair.

If we take the cross complementary sequence pair $(C_0(z), S_0(z))$ and $(C_1(z), S_1(z))$ then:

1. the sequence represented by $C_0(z)$ followed by $W_0 \geq \tau_{\max} - 1$ zeros followed by the sequence represented by $S_0(z)$;

2. the sequence represented by $C_1(z)$ followed by $W_0 \geq \tau_{\max} - 1$ zeros followed by the sequence represented by $S_1(z)$;

have ideal correlation values for the given value of $d = \tau_{\max} - 1$ (zero aperiodic cross-correlation and zero aperiodic autocorrelation when $\tau \neq 0$).

Using a $[p \times p]$ Hadamard matrix H (where $H = [h_{i,j}]$), and a cross complementary sequence pair $(C_0(z), S_0(z))$ and $(C_1(z), S_1(z))$ it is possible to create $2p$ LS codewords of length $L = 2pN + W_0$. Let $\pi = (\pi_1, \pi_2, \ldots, \pi_p)$ be a binary vector of length p with binary complement $\pi^* = (\pi_1^*, \pi_2^*, \ldots, \pi_p^*)$. Then let the $2p$ sequences of an LS code for $1 \leq k \leq p$ be represented by [21]:

$$\begin{aligned} G_k(z) &= \textstyle\sum_{i=1}^{p} h_{k,i} \left[C_{\pi_i}(z) + z^{pN+W_0} S_{\pi_i}(z) \right] z^{(i-1)N} \\ G_{k+p}(z) &= \textstyle\sum_{i=1}^{p} h_{k,i} \left[C_{\pi_i^*}(z) + z^{pN+W_0} S_{\pi_i^*}(z) \right] z^{(i-1)N}. \end{aligned} \tag{2.6}$$

With the choice of $W_0 = N - 1$, the off-peak aperiodic autocorrelation and aperiodic cross-correlation is zero for $|\tau| \leq W_0$. $2p$ sequences of length $2pN + 2(N-1)$ are constructed with zero correlation window $\tau_{\max} - 1 = W_0$.

For example, consider a Golay pair and its mate of length $N = 16$:

$$C_0(z) = 1 + z + z^2 - z^3 + z^4 + z^5 - z^6 + z^7 + z^8 + z^9 + z^{10} - z^{11} - z^{12} - z^{13} + z^{14} - z^{15}$$
$$S_0(z) = 1 + z + z^2 - z^3 + z^4 + z^5 - z^6 + z^7 - z^8 - z^9 - z^{10} + z^{11} + z^{12} + z^{13} - z^{14} + z^{15}$$

$C_1(z) = -1+z-z^2-z^3-z^4+z^5+z^6+z^7+z^8-z^9+z^{10}+z^{11}-z^{12}+z^{13}+z^{14}+z^{15}$
$S_1(z) = -1+z-z^2-z^3-z^4+z^5+z^6+z^7-z^8+z^9-z^{10}-z^{11}+z^{12}-z^{13}-z^{14}-z^{15}$

Select $W_0 = N - 1 = 15$ and $\pi = (0, 1)$ (so that its complement $\pi^* = (1, 0)$). Let the Hadamard matrix be a 2×2 Sylvester type matrix so that 4 codewords of length $L = (2 \times 2 \times 16) + 30 = 94$ with duty ratio 0.68 are created. The LS sequences constructed are:

$$
\begin{array}{lccccccc}
G_1(z) = & 0000000 & C_0(z) & C_1(z) & 000000000000000 & S_0(z) & S_1(z) & 00000000 \\
G_2(z) = & 0000000 & C_0(z) & -C_1(z) & 000000000000000 & S_0(z) & -S_1(z) & 00000000 \\
G_3(z) = & 0000000 & C_1(z) & C_0(z) & 000000000000000 & S_1(z) & S_0(z) & 00000000 \\
G_4(z) = & 0000000 & C_1(z) & -C_0(z) & 000000000000000 & S_1(z) & -S_0(z) & 00000000
\end{array}
$$

The zero correlation window has $\tau_{\max} = 16$. If a bigger Hadamard matrix was used to generate an LS sequence then more codewords would be created but the code would be longer. For example, with the same cross complementary sequence pair as above and the use of a 32×32 Hadamard matrix 64 codewords will be generated. The length of each LS codeword increases to 1054 with duty ratio 0.97.

2.3 LS codes with internal padding

A generalized construction of LS codes with further padding has been presented by Sanusi *et al.* [17] and overcomes the limit on the number of codewords given by Tang, Fan and Matsufuji [24]. Tang, Fan and Matsufuji derived a bound for the aperiodic correlation for a code with u codewords, length n and $0 \leq \mid \tau \mid < T$ which implies that $(u-1)T \leq n-1$. The modification by Sanusi *et al.* [17] doubles the number of codewords produced with a small increase in the length of the LS code and with a limited degradation in the correlation properties. The modification will be described in detail in this section.

Following from the LS code construction in section 2.2, suppose the length N of the Golay pair is halved and the size p of the Hadamard matrix is doubled. The new Golay pair has length $\mathcal{N} = N/2$. Then $4p$ LS sequences of the same length can be constructed. Good correlation properties are necessary for $|\tau| \leq \min(2\mathcal{N} - 1, W_0)$. However, using the construction of equation 2.6, large values of the aperiodic correlation can be observed for $|\tau| = \mathcal{N}$, as the number of exact coincidences of Golay pair components can be large.

In the modified construction by Sanusi *et al.* [17] it is necessary to define a set of *internal padding lengths* $\{L_1, L_2, L_3, \ldots, L_{\mathcal{P}-1}\}$ such that $\mathcal{P} = 2p$, and also an ordering for these lengths. Assume that the implied ordering $L_1, L_2, L_3, \ldots, L_{\mathcal{P}-1}$ is chosen. Then cumulative padding lengths are $Q_1 = 0$,

$Q_i = \sum_{j=1}^{i-1} L_j$ $(2 \leq i \leq \mathcal{P})$. Once the ordering is chosen, internal padding is defined by an *internal padding vector* $(L_1, L_2, L_3, \ldots, L_{\mathcal{P}-1})$. An internal padding vector with internal padding lengths $\{L_1, L_2, L_3, \ldots, L_{\mathcal{P}-1}\}$ is referred to as *feasible.*

Suppose $H = [h_{i,j}]$ is a $\mathcal{P} \times \mathcal{P}$ Hadamard matrix, and use a cross complementary sequence pair given by $(C_0(z), S_0(z))$ and $(C_1(z), S_1(z))$. Let $\pi = (\pi_1, \pi_2, \ldots, \pi_{\mathcal{P}})$ be a binary vector of length $\mathcal{P}$ and let π^* be the complement of π. Then the $2\mathcal{P}$ sequences of an LS code with internal padding for $1 \leq k \leq \mathcal{P}$ can be represented by:

$$\begin{aligned} G_k(z) &= \sum_{i=1}^{\mathcal{P}} h_{k,i} \left[C_{\pi_i}(z) + z^{\mathcal{P}N+W_0+Q_{\mathcal{P}}} S_{\pi_i}(z)\right] z^{(i-1)N+Q_i} \\ G_{k+\mathcal{P}}(z) &= \sum_{i=1}^{\mathcal{P}} h_{k,i} \left[C_{\pi_i^*}(z) + z^{\mathcal{P}N+W_0+Q_{\mathcal{P}}} S_{\pi_i^*}(z)\right] z^{(i-1)N+Q_i}. \end{aligned} \tag{2.7}$$

Note that if all $L_j = 0$ the construction reduces to that of equation 2.6. Now, in order to prevent a small duty ratio, distinct internal padding lengths are repeated a small number of times. This involves considering transitions and non-transitions in the elements of vectors π and π^*. A *transition* occurs when:

1. one Golay pair C member is followed by a different Golay pair C member, i.e. C_0 followed by C_1 or C_1 followed by C_0;

2. one Golay pair S member is followed by a different Golay pair S member, i.e. S_0 followed by S_1 or S_1 followed by S_0.

A *non-transition* occurs when:

1. one Golay pair C member is followed by the same Golay pair C member, i.e. C_0 followed by C_0 or C_1 followed by C_1;

2. one Golay pair S member is followed by the same Golay pair S member, i.e. S_0 followed by S_0 or S_1 followed by S_1.

For each transition, the corresponding increase in the aperiodic correlation is zero (from equation 2.4) for two codewords generated by π or two codewords generated by π^*. For each non-transition, the corresponding increase in the aperiodic correlation is zero (from equation 2.4) for one codeword generated by π and one codeword generated by π^*. Note that the total number of transitions and non-transitions is equal to $\mathcal{P} - 1$. Let (Tr,NTr) denote the number of transitions and the number of non-transitions. For example, three cases for the value of (Tr,NTr) can be considered when lengths are repeated four times.

Case A: $(Tr,NTr) = (\frac{\mathcal{P}}{2}, \frac{\mathcal{P}}{2} - 1)$.

In this case vector π is chosen with $\frac{\mathcal{P}}{2}$ transitions and $\frac{\mathcal{P}}{2}-1$ non-transitions.

$$\pi = (0110011001100110\ldots1001100110)$$

Transitions are assigned padding lengths

$$\{0,0,1,1,2,2,\ldots,\mathcal{P}/4-2,\mathcal{P}/4-2,\mathcal{P}/4-1,\mathcal{P}/4-1\}$$

(in some order) and non-transitions are assigned padding lengths

$$\{0,0,1,1,2,2,\ldots,\mathcal{P}/4-2,\mathcal{P}/4-2,\mathcal{P}/4-1\}.$$

Some consecutive Golay pairs in distinct codewords could give a nonzero contribution to the aperiodic correlation (referred to here as a *coincidence*) for some values of τ with $\mathcal{N} \leq |\tau| < 2\mathcal{N}$. If we let $W_0 = 2\mathcal{N}-1$ then the aperiodic cross-correlation and off-peak aperiodic autocorrelation is 0 for $0 \leq| \tau |\leq \mathcal{N}-1$ and $\mathcal{N}+\mathcal{P}/4 \leq \tau \leq 2\mathcal{N}-1$. The aperiodic correlations have nonzero (but still fairly small) values in the interval $\mathcal{N} \leq \tau \leq \mathcal{N}+\mathcal{P}/4-1$ as a result of the coincidences.

Case B: $(Tr,NTr) = (\mathcal{P}-1,0)$.

$$\pi = (01010101010101\ldots10101)$$

In this case all internal padding lengths are of necessity assigned to transitions. If $W_0 = 2\mathcal{N}-1$ the aperiodic correlation for any two distinct codewords both using π or both using π^* is 0 for $0 \leq| \tau |\leq 2\mathcal{N}-1$. Otherwise the aperiodic correlation is 0 for $0 \leq| \tau |\leq \mathcal{N}-1$ and $\mathcal{N}+\mathcal{P}/4 \leq \tau \leq 2\mathcal{N}-1$. It gives some nonzero (but still fairly small) values for $\mathcal{N} \leq \tau \leq \mathcal{N}+\mathcal{P}/4-1$.

Case C: $(Tr,NTr) = (\frac{3\mathcal{P}}{4},\frac{\mathcal{P}}{4}-1)$.

$$\pi = (0101101001011010011011010\ldots0101101001011010)$$

In this case three instances of each internal padding length are assigned to transitions and one to non-transitions. If $W_0 = 2\mathcal{N}-1$ the aperiodic correlation for any two codewords is 0 for $0 \leq| \tau |\leq \mathcal{N}-1$ and $\mathcal{N}+\mathcal{P}/4 \leq \tau \leq 2\mathcal{N}-1$. It gives some nonzero (but still fairly small) values for $\mathcal{N} \leq \tau \leq \mathcal{N}+\mathcal{P}/4-1$.

In this work LS codes with internal paddings will be used. Here a standard example is defined which will be used throughout most of this thesis. Some alternative examples are considered in chapter 11.

Example 1 *A Golay pair of length* $\mathcal{N} = N/2 = 16$ *(such that* $W_0 = 2\mathcal{N}-1 = 31$*) and a* 32×32 *Hadamard matrix are used to create a standard example. The*

maximum number of repetitions for the internal padding lengths is 4 (so that $Q_{\mathcal{P}}$ is 105). The length of the LS code constructed is then 1296. Case A with (Tr,NTr) = (16,15) is used in this standard example. The peak aperiodic correlation which occurs within the interval $(16 \leq |\tau| \leq 23)$ *is 64.*

The length of 1296 is perhaps the maximum length that might be considered useful. The code is designed for quasi-synchronous operation with $\tau_{\max} = 32$. *This value is larger than is sometimes considered in the literature, but allows quasi-synchronous operation to be contemplated in a wider variety of circumstances.*

Chapter 3

The distribution of aperiodic correlations between generations as a result of permutations

In this chapter the distribution of aperiodic correlations between the codewords assigned to a single user in different generations is discussed. The new generations are obtained simply by applying permutations.

3.1 Permutations of positions of the internal padding vector

In this section results are presented of experiments using random permutations of the positions of the internal padding vector. Firstly, a large number of randomly generated internal padding vectors were created by permuting the positions of an original internal padding vector. These paddings were then used with a fixed Golay pair/mate combination and a fixed Hadamard matrix to create LS codes for 50 different generations. The peak aperiodic cross-correlations between all pairs of generations over the interval $|\tau| < \tau_{\max}$ were computed for a single user. The results for this single user are given in figure 3.1. The correlations arise from exact coincidences of Golay pairs and so take only a limited number of distinct values. The results show that this permutation already produces moderately good results. With just randomly generated permutations of the positions of the internal padding, the peak value is only 352 (0.34375 when normalized to [0,1]). With randomly generated permutations there is a possibility that the same permutation of the internal padding lengths might be generated twice. The likelihood of this occurring in fifty generations is very small, but over a larger number of generations it must be considered.

3.2 Row permutations

This section considers random permutations of the rows of the Hadamard matrix. A row permutation is simply a random reordering of the original set of codewords. In this way the codewords assigned to users are permuted. As there are only 64 codewords, they may be reused. Therefore the peak normalized aperiodic correlation between generations may be 1.0, otherwise the peak aperiodic correlation is the peak aperiodic cross-correlation for a single LS code. Results for a single user are given in figure 3.2.

3.3 Column permutations

This section considers random permutations of the columns of the Hadamard matrix. A column permutation is simply a random reordering of the columns of the Hadamard matrix. Results for a single user are given in figure 3.3. The normalized aperiodic cross-correlation values are generally low (mostly below 0.3) but the peak value can be above 0.5.

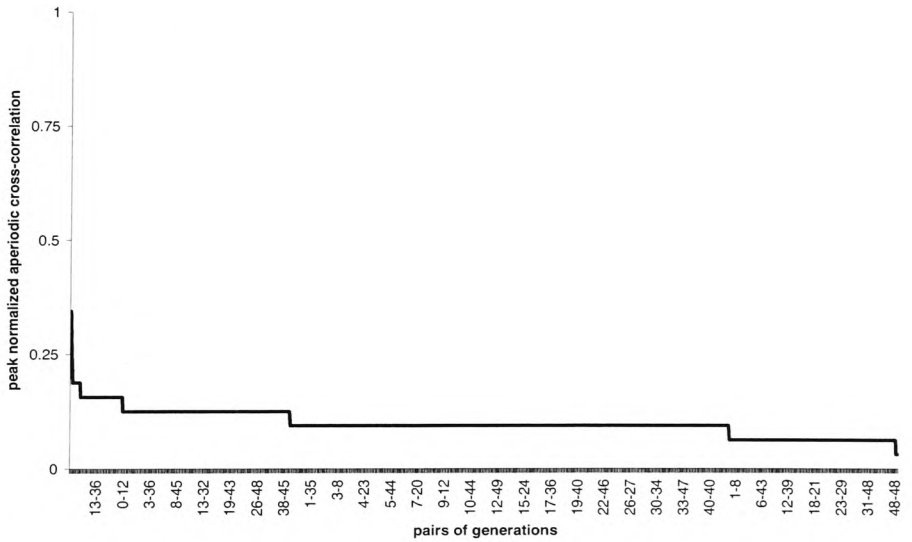

Figure 3.1: *The peak normalized aperiodic cross-correlation for a single user (LS code of length 1296 and 1024 ± 1's) using a random internal padding permutation to calculate the next generation. The aperiodic cross-correlations are calculated with $|\tau| < \tau_{\max}$. The pairs of generations have been sorted in decreasing order of correlation.*

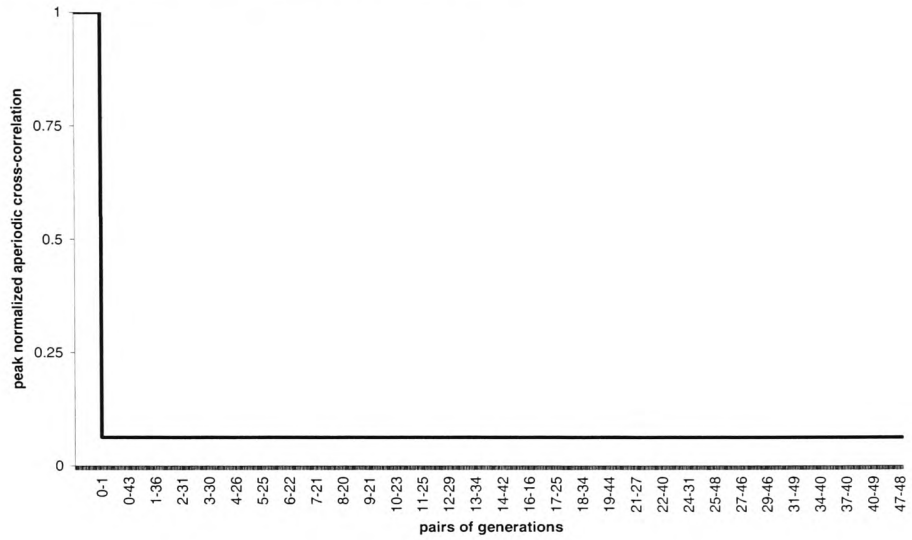

Figure 3.2: *The peak normalized aperiodic correlation for a single user (LS code of length 1296 and 1024 ± 1's) using a random row permutation of the Hadamard matrix. The aperiodic correlations are calculated with $|\tau| < \tau_{\max}$. The pairs of generations have been sorted in decreasing order of correlation.*

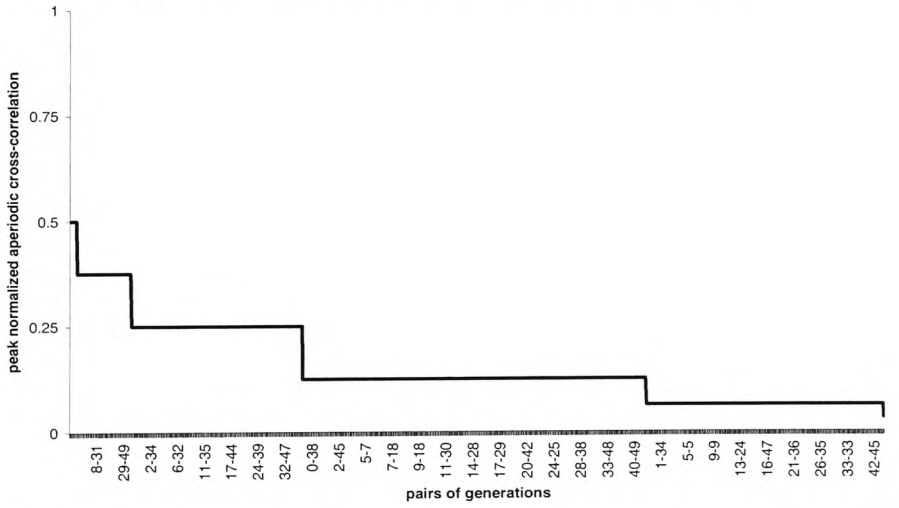

Figure 3.3: *The peak normalized aperiodic cross-correlation for a single user (LS code of length 1296 and 1024 ± 1's) using a random column permutation of the Hadamard matrix. The aperiodic cross-correlations are calculated with $|\tau| < \tau_{\max}$. The pairs of generations have been sorted in decreasing order of correlation.*

3.4 Summary of the individual distributions

This section summarises individually the distributions of correlations arising from the above random permutations in the LS code construction. Specifically, the

internal padding may be permuted as in section 3.1, the rows of the Hadamard matrix may be permuted as in section 3.2, or the columns of the Hadamard matrix may be permuted as in section 3.3. The results for the peak normalized aperiodic correlation of a single user over 50 generations are given in table 3.1. These can be used to assess the relative merits of each type of permutation. Table 3.1 shows the peak modulus, mean modulus and standard deviation for a single user over all 1225 pairs of generations. In the table *IP* denotes a random permutation of the internal padding vector, *Row* is a permutation of the rows of the Hadamard matrix and *Col* is a permutation of the columns of the Hadamard matrix.

Note that for column permutations a non-normalized Hadamard matrix should be used; if the matrix is normalized (or of Sylvester type), column permutations have no effect for the first codeword.

It can be seen from table 3.1 that permutations of the internal padding alone already give reasonably good results, but cannot guarantee that peak correlation does not exceed 0.25. Column permutations are not as good in terms of the peak correlation, but are equally good in terms of the mean. Permutations of the rows of the Hadamard matrix give a smaller mean as the correlations are just those of the LS code itself. However, some codewords will be reused over 50 generations, so the peak aperiodic correlation is equal to the autocorrelation peak. Thus row permutations should certainly not be used on their own.

Component permuted	*Peak*	*Mean*	*standard dev.*
IP	0.344	0.156	0.038
Row	1	0.084	0.140
Column	0.625	0.151	0.097

Table 3.1: *Correlation results over fifty generations for a single user.*

Combinations of the permutations have also been investigated. The results of these combinations do not substantialy improve the results of using the permutations of elements of the internal padding vector, and are therefore not presented.

3.5 Permutations of codewords using Latin squares

Although row permutations are an option, it is desirable to avoid repetition of codewords for a single user in different generations. The codewords assigned to

individual users should be permuted to obtain new generations in such a way that no user can keep the same codeword. This can be achieved using Latin squares. A Latin square of order n is an $n \times n$ array containing symbols from some alphabet of size n, arranged such that each element appears exactly once in each row and in each column.

$$ABCD$$
$$DABC$$
$$CDAB$$
$$BCDA$$

The Latin square above has 4 elements and 4 ways to arrange them without repetition in any row or column. The same can be done for any size square. A Latin square of codewords for n users creates a maximum of n generations with no element repeated. This means that it is possible with the 64 LS codewords with $p = 32$, $N = 16$ to construct 64 generations of codewords with the set of cross-correlation values between generations for a single user being the same as the set of values given by the LS code itself. Results for any single user are shown in figure 3.4.

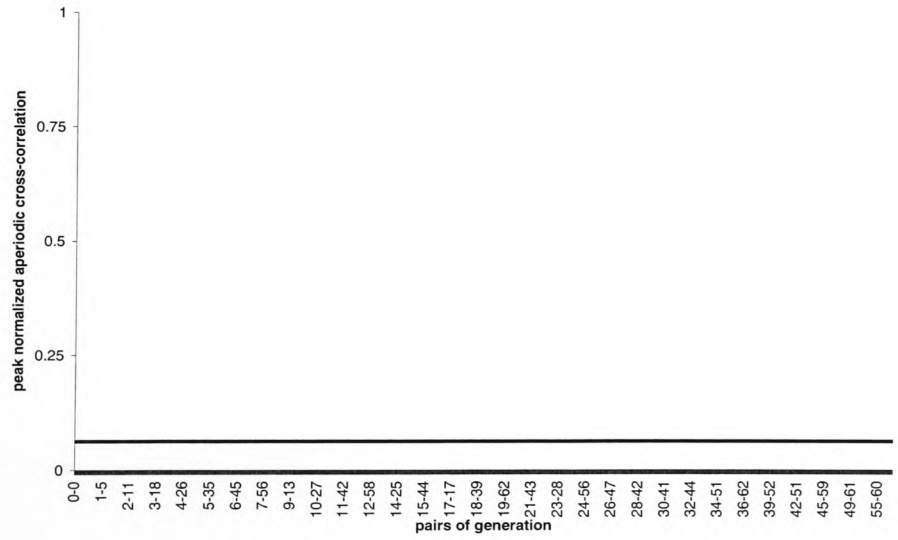

Figure 3.4: *The peak normalized aperiodic cross-correlation for a single user (LS code of length 1296 and 1024 ± 1's) using a Latin square to calculate the generations. The aperiodic cross-correlations are calculated with $|\tau| < \tau_{\max}$. The pairs of generations have been sorted in decreasing order of correlation.*

3.6 Summary

As already noted, permutations can already give reasonably good results, as long as the permutations include permutations of the elements of the internal padding vector. However, the normalized aperiodic correlations are not always below the value of 0.25 which has been used as a typical upper bound in this work. In subsequent chapters, ways in which sets of Golay pairs, sets of internal padding vectors and sets of Hadamard matrices can be selected to obtain better correlation results will be described.

Chapter 4

Sets of Golay pairs with favourable correlation properties

In chapter 2 it was noted that there are Golay pairs of all lengths

$$2^{\alpha}10^{\beta}26^{\gamma}, \quad \alpha, \beta, \gamma \geq 0.$$

In this chapter only Golay pairs of length a power of 2 are used.

Golay pairs have six operations that establish equivalence classes of different Golay pairs of the same length.

1) interchanging the sequences in a pair,
2) reversing the order of the first,
3) reversing the order of the second,
4) changing the sign of the first,
5) changing the sign of the second,
6) changing the sign of alternate entries in both the first and the second.

It is easy to see that each operation turns a Golay pair into a Golay pair. If N is large enough, 64 Golay pairs can be obtained in this manner.

A construction given in [5] explicitly determines $2^{m+2}m!/2$ vectors over $\mathbf{Z}_2$ corresponding to Golay pairs of length 2^m. The construction is as follows:

Definition 10 *Let $x_1, x_2, ..., x_m$ be a set of m binary variables. Let c, c' and $c_k \in \mathbf{Z}_2$ (k = 1,2,...,m) and π be a permutation of the symbols $\{1, 2, ..., m\}$. Define a Boolean function*

$$f(x_1, x_2, ..., x_m) = \sum_{k=1}^{m-1} x_{\pi(k)}x_{\pi(k+1)} + \sum_{k=1}^{m} c_k x_k$$

where all arithmetic is mod 2. $f(x_1, x_2, ..., x_m)$ *defines a sequence by evaluating the function for all binary vectors in binary order*

$$00...00, 00...01, ..., 00...10, ..., 11...11.$$

Then for all choices of π,c,c',$\{c_k\}$,*any sequence in the set*

$$A = \{f + c, f + (x_{\pi(1)} + x_{\pi(m)}) + c\}$$

corresponds to a Golay pair of length 2^m *with any sequence in the set*

$$B = \{f + x_{\pi(1)} + c', f + x_{\pi}(m) + c'\}.$$

Using this construction, all vectors of length a power of 2 can be created and converted to Golay pairs with elements from {-1,+1} using the mapping $g : x \mapsto (-1)^x$.

m	2	3	4	5	6
T	16	96	768	7680	92160

Table 4.1: *The number* T *of Golay pairs of length* 2^m *created using the construction found in [5]. Golay pairs* (A, B) *and* $(-A, -B)$ *are only counted once.*

Note that for a Golay pair (C_0, S_0) with mate (C_1, S_1), knowledge of C_0, C_1 implies knowledge of S_0, S_1 to within a $\pm$ sign.

Given a set of such Golay pairs, a set of LS codes can be constructed using the same Hadamard matrix, internal padding vector and π vector, as in chapter 3. For each Golay pair and mate generated there is an LS code. For two codes (used in two different generations) the codeword assigned to a single user (same row of the Hadamard matrix, with π or π^*) can be used to compute the peak aperiodic correlations. Now construct a graph as follows. Each vertex corresponds to a code in the set. These codes in the set are used for the different generations of an evolution. Two vertices are joined if the maximum of the peak normalized aperiodic cross-correlations for all users is less than or equal to a threshold σ. A maximum clique in a graph G is the largest complete subgraph contained in G. A maximum clique in G can be computed using the algorithm of Carraghan and Pardalos [3]. This provides a way of finding codes for the largest number of generations possible with normalized correlations at most a given peak value σ. Note that the correlation criterion is different from that used in chapter 3 which gave results for a single codeword (user) rather than the maximum over all codewords (users). The largest complete graphs with maximum peak normalized aperiodic cross-correlation σ can be seen in table 4.2. To ensure an effective level of security is maintained the peak normalized

aperiodic cross-correlation value computed for each user and the peak taken over all users should be small. Reasonable compromise values for σ, giving a reasonable number of generations are 0.282 ($N = 8$) and 0.274 ($N = 16$). Reducing these values to 0.25 gives much smaller cliques.

N	μ	σ	*Max. no. generations*
8	512	1	192
8	512	0.5	52
8	512	0.282	13
8	512	0.25	8
8	512	0	2
16	1024	1	384
16	1024	0.5	196
16	1024	0.274	22
16	1024	0.25	8
16	1024	0	2

Table 4.2: *The maximum number of generations for various levels of peak normalized aperiodic cross-correlation for the LS codes constructed using the clique search for Golay pairs. N is the length of the Golay pair and μ is the number of ± 1's in each codeword.*

In figure 4.1 detailed results can be seen for 22 Golay pairs with $N = 16$ selected with their mates from the clique and used to create 22 different LS codes of length 1296 for a single user. The 22 codes have very low peak normalized aperiodic cross-correlation values (≤ 0.274). As already seen in table 4.2, only 8 generations are possible with the threshold at 0.25 (shown for a single user in figure 4.2).

This chapter shows that if a simple clique search is used it is possible to find enough Golay pairs (and therefore enough generations) with a chosen threshold σ below 0.3 to allow the Golay pairs to evolve usefully. In chapter 7 a method of *guaranteeing* the maximum peak normalized aperiodic correlation without consideration of the Hadamard matrix or the internal padding vector will be presented.

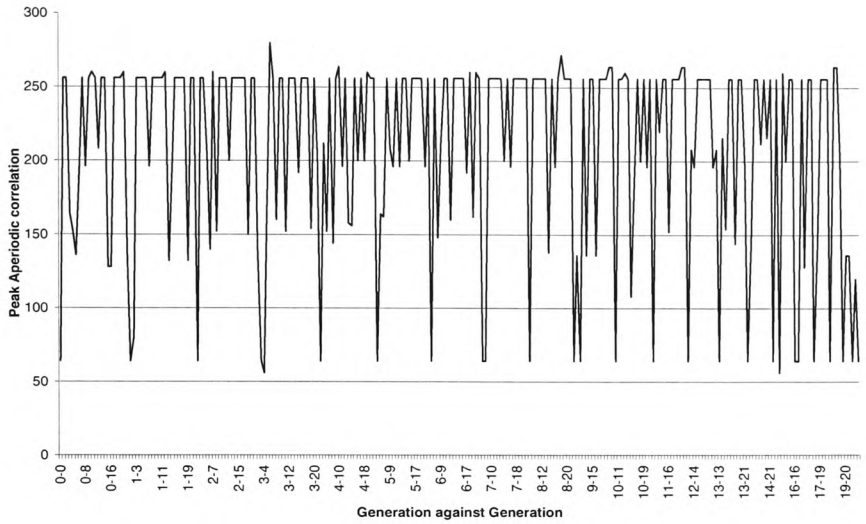

Figure 4.1: *The peak aperiodic cross-correlation for LS codes produced from 22 different Golay pairs and their mates with* $N = 16$*, with maximum peak normalized aperiodic cross-correlation 0.274.*

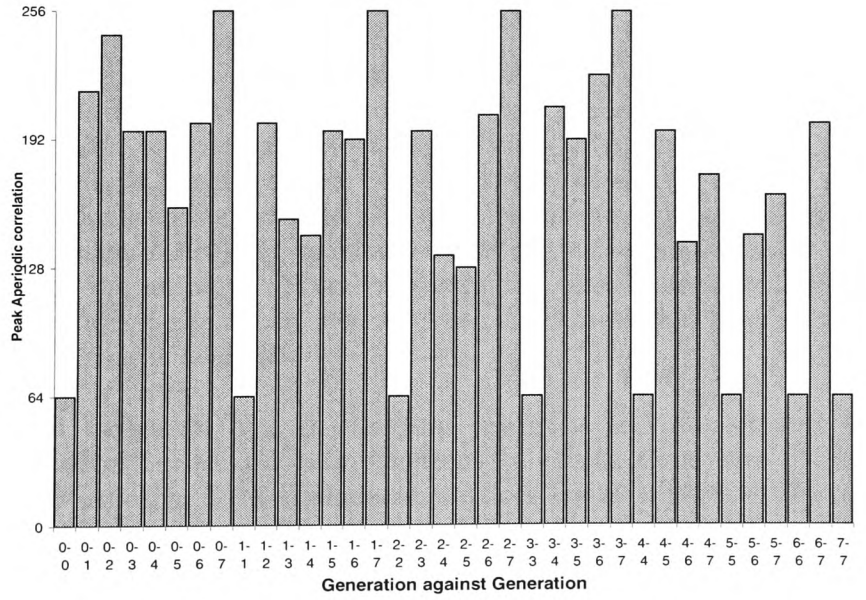

Figure 4.2: *The peak aperiodic cross-correlation for LS codes produced from 8 different Golay pairs and their mates with* $N = 16$*, with maximum peak normalized aperiodic cross-correlation 0.25.*

Chapter 5

Bent functions, almost bent functions and Hadamard matrices

One of the possibilities for evolving the code is to construct a set of distinct Hadamard matrices. Then different generations of the code can use different Hadamard matrices. In order to ensure that the correlation criterion between codewords from distinct generations is satisfied, this needs to be done in such a way that the normalized inner product between two rows of two distinct matrices is bounded above. The construction given in [29] and [25] is used here, together with new constructions for m even based on Gold and Gold-like codes. Once the set of Hadamard matrices is available, a set of LS codes can be obtained using the construction of chapter 2 with each Hadamard matrix in turn. A bound on the normalized inner product between two rows of two distinct Hadamard matrices in the set then leads to the same bound on the normalized aperiodic correlation (when $\tau = 0$) of two codewords in distinct codes of the set. The code consisting of the union of all codewords in the set of codes still has the zero correlation zone except when $\tau = 0$, when the correlation is nonzero but the correlation bound is available.

Here it will be shown how to use *bent functions* and *almost bent* functions and other second order Boolean functions to create these sets of Hadamard matrices. Specifically, if the Hadamard matrices are $2^m \times 2^m$ then the maximum normalized inner product of two rows of distinct matrices will be $2^{-(m/2)}$ (m even) or $2^{-((m-1)/2)}$ (m odd).

Bent and almost bent functions are $\{0,1\}$ Boolean functions $f(x_1, x_2, \ldots, x_m)$ of m $\{0,1\}$ Boolean variables. Arrange the set of all values of the vector $(x_1, x_2, \ldots, x_m)$ in binary order ranging from $(0,0,0,\ldots,0)$ to $(1,1,1,\ldots,1)$. These values index the columns of a matrix, and also define a vector

$\hat{\mathbf{f}} = (f(0,0,\ldots,0),\ldots,f(1,1,\ldots,1))$. For a bent or almost bent function f_i, and an initial $2^m \times 2^m$ Hadamard matrix H in Sylvester form, the columns of H are multiplied by the values of the $\{-1,+1\}$ function $\tilde{f}_i(x_1,x_2,\ldots,x_m) = (-1)^{f_i(x_1,x_2,\ldots,x_m)}$ to obtain the ith matrix in the set. (Note that the nonzero vectors indexing the columns can alternatively be the vector representations of the powers of a primitive element α in an appropriate finite field. In this way the usual extended cyclic structure of the (0,1) codes involved can be seen. Thus, for example, with this extended cyclic structure it can be checked that the nonzero codewords of an extended m-sequence give a Hadamard matrix under the mapping of the elements $g : x \mapsto (-1)^x$. Here, however, the first vector ordering will be used.)

Extend the definition of $g : x \mapsto (-1)^x$ to a mapping of vectors: $g(\hat{\mathbf{f}})$ is the vector obtained by mapping the component x_i of $\hat{\mathbf{f}}$ to $g(x_i)$. Thus g maps $\{0,1\}$ vectors to $\{-1,+1\}$ vectors.

The notation can be summarised as follows:

1. f is a $\{0,1\}$ Boolean function;
2. $\tilde{f}$ is a $\{-1,+1\}$ Boolean function;
3. $\hat{\mathbf{f}}$ is the vector corresponding to f;
4. $g(\hat{\mathbf{f}})$ is the vector corresponding to $\tilde{f}$.

Definition 11 *The Boolean function f of m variables is defined in [29] as bent (for m even) if the inner product between $g(\hat{\mathbf{f}})$ and a row of H has magnitude $2^{m/2}$ and as almost bent (for m odd) if the magnitude of the inner product between $g(\hat{\mathbf{f}})$ and a row of H is at most $2^{(m+1)/2}$.*

In [29] a formula is given for a set of 2^{m-1} bent function when m is even, which then leads to a set of 2^{m-1} Hadamard matrices with maximum correlation $2^{m/2}$ between any pairs of rows of distinct matrices. Similarly, a formula is given for a set of 2^m almost bent functions when m is odd, which then leads to a set of 2^m Hadamard matrices with maximum correlation $2^{(m+1)/2}$ between any pairs of rows of distinct matrices. These functions and some examples are given below. Note that it is possible to consider the rows of the initial Hadamard matrix (converted to a set of $\{0,1\}$ vectors using g^{-1}) as a subcode of a first order Reed-Muller code [14]. Similarly, the bent (almost bent) functions, when converted to vectors, become words in second order Reed-Muller codes. For m even the bent functions also have the property that they correspond to words of a Kerdock code [14] and for m odd the almost bent functions correspond to words of an extended Gold code [18].

The functions can be computed in Maple[1] using binary arithmetic and Maple simplifications. The specific simplifications necessary are:

1. expanding the functions;
2. simplifications modulo the polynomial generating the appropriate finite field;
3. simplifications $x^i \equiv x$ appropriate to Boolean functions.

In a finite field $GF(2^m)$ the trace function is defined by

$$\mathrm{Tr}_m(x) = \sum_{i=0}^{m-1} x^{2^i}.$$

The trace function is a linear function ($\mathrm{Tr}_m(\beta_1 + \beta_2) = \mathrm{Tr}_m(\beta_1) + \mathrm{Tr}_m(\beta_2)$ for $\beta_1, \beta_2 \in GF(2^m)$) and satisfies $\mathrm{Tr}_m(\beta^2) = \mathrm{Tr}_m(\beta)$ for all $\beta \in GF(2^m)$ [14].

5.1 Construction using bent functions in the case of m even.

Let $\xi_2, ..., \xi_m$ be a basis for $GF(2^{m-1})$ and $x = \sum_{i=2}^{m} x_i \xi_i$ be an expression for an element x in terms of this basis. Let α be a primitive element of $GF(2^{m-1})$. In the examples below $\xi_2 = 1$, $\xi_3 = \alpha$, ..., $\xi_m = \alpha^{m-2}$. Let $\sigma_1(x) = \mathrm{Tr}_{m-1}(x)$ and $\sigma_2(x) = \sum_{i=1}^{(m-2)/2} \mathrm{Tr}_{m-1}(x^{1+2^i})$. Then for any $\gamma \in GF(2^{m-1})$ a bent function is defined in [29, 25] by

$$h_\gamma(x_1, x_2, ..., x_m) = x_1 \sigma_1(\gamma x) + \sigma_2(\gamma x)$$

The sets of 2^{m-1} bent functions can be computed as follows.
i) The case $\mathbf{m = 4}$. Here the Galois field $GF(2^{m-1})$ is constructed using any irreducible polynomial of degree 3. For the results below $\alpha^3 + \alpha + 1 \equiv 0$ is

[1] http://www.maplesoft.com/

selected:

$$\begin{aligned}
h_0 &= 0 \\
h_1 &= x_2 + x_3x_4 + x_3 + x_4 + x_1x_2 \\
h_\alpha &= x_2x_3 + x_2 + x_3x_4 + x_3 + x_1x_4 \\
h_{\alpha^2} &= x_2x_3 + x_2x_4 + x_2 + x_3x_4 + x_4 + x_1x_3 \\
h_{\alpha^3} &= x_2x_3 + x_2x_4 + x_3 + x_1x_2 + x_1x_4 \\
h_{\alpha^4} &= x_2x_4 + x_2 + x_3x_4 + x_1x_4 + x_1x_3 \\
h_{\alpha^5} &= x_2x_3 + x_4 + x_1x_2 + x_1x_3 + x_1x_4 \\
h_{\alpha^6} &= x_2x_4 + x_3 + x_4 + x_1x_2 + x_1x_3
\end{aligned}$$

ii) The case $\mathbf{m = 6}$. Here the Galois field $GF(2^{m-1})$ is constructed using any irreducible polynomial of degree 5. For the results below $\alpha^5 + \alpha^2 + 1 \equiv 0$ is selected:

$$\begin{aligned}
h_0 &= 0 \\
h_1 &= x_6x_5 + x_6x_4 + x_6x_2 + x_2x_3 + x_2x_5 + x_5x_3 + x_4 + x_3x_4 + x_1x_4 + x_1x_6 \\
h_\alpha &= x_6x_4 + x_6 + x_5x_3 + x_4x_5 + x_4x_2 + x_2x_3 + x_3 + x_1x_5 + x_1x_3 + x_1x_6 \\
h_{\alpha^2} &= x_6x_5 + x_3x_6 + x_6x_4 + x_6x_2 + x_6 + x_3x_4 + x_4x_2 + x_2 + x_5x_3 + x_5 \\
&\quad + x_1x_5 + x_1x_4 + x_1x_2 \\
h_{\alpha^3} &= x_3x_6 + x_6x_2 + x_4x_2 + x_5 + x_5x_3 + x_2x_3 + x_2x_5 + x_4x_5 + x_4 + x_1x_3 + x_1x_4 \\
h_{\alpha^4} &= x_6x_5 + x_3x_6 + x_6x_2 + x_3 + x_4x_2 + x_2x_5 + x_4 + x_3x_4 + x_1x_3 + x_1x_6 + x_1x_2 \\
h_{\alpha^5} &= x_6x_5 + x_3x_6 + x_6x_4 + x_6x_2 + x_2x_5 + x_3 + x_2 + x_4x_5 + x_2x_3 + x_1x_5 \\
&\quad + x_1x_6 + x_1x_2 \\
h_{\alpha^6} &= x_6x_2 + x_6 + x_2x_5 + x_5x_3 + x_4x_5 + x_2 + x_3x_4 + x_1x_5 + x_1x_4 + x_1x_6 \\
h_{\alpha^7} &= x_6 + x_6x_2 + x_6x_5 + x_2x_3 + x_5 + x_3x_4 + x_4x_2 + x_1x_5 + x_1x_3 + x_1x_4 + x_1x_6 \\
h_{\alpha^8} &= x_6x_5 + x_3x_6 + x_6 + x_5 + x_4x_5 + x_4 + x_2x_3 + x_1x_5 + x_1x_3 + x_1x_4 \\
&\quad + x_1x_6 + x_1x_2 \\
h_{\alpha^9} &= x_6x_4 + x_6x_2 + x_6 + x_4x_5 + x_2x_5 + x_3x_4 + x_5 + x_4 + x_3 + x_1x_5 + x_1x_3 \\
&\quad + x_1x_4 + x_1x_2 \\
h_{\alpha^{10}} &= x_6x_2 + x_6 + x_3 + x_5 + x_2 + x_4 + x_5x_3 + x_2x_3 + x_3x_4 + x_1x_3 + x_1x_4 \\
&\quad + x_1x_2 \\
h_{\alpha^{11}} &= x_6x_5 + x_2 + x_4x_2 + x_4 + x_5 + x_3 + x_2x_3 + x_1x_3 + x_1x_2 \\
h_{\alpha^{12}} &= x_6 + x_3x_6 + x_6x_2 + x_2 + x_4x_5 + x_4 + x_3 + x_1x_6 + x_1x_2 \\
h_{\alpha^{13}} &= x_6x_5 + x_6x_4 + x_2x_5 + x_2 + x_5 + x_3 + x_3x_4 + x_1x_5 + x_1x_6 \\
h_{\alpha^{14}} &= x_6x_5 + x_6x_4 + x_6x_2 + x_6 + x_2 + x_2x_3 + x_4x_5 + x_5x_3 + x_4 + x_1x_5 + x_1x_4
\end{aligned}$$

$$
\begin{aligned}
h_{\alpha^{15}} &= x_6x_4 + x_6x_2 + x_3 + x_5x_3 + x_4x_5 + x_3x_4 + x_4x_2 + x_5 + x_1x_3 + x_1x_4 + x_1x_6 \\
h_{\alpha^{16}} &= x_3x_6 + x_6x_4 + x_6x_2 + x_6 + x_2x_3 + x_3x_4 + x_2 + x_4 + x_4x_2 + x_5x_3 + x_1x_5 \\
&\quad + x_1x_3 + x_1x_6 + x_1x_2 \\
h_{\alpha^{17}} &= x_3x_6 + x_6 + x_2x_5 + x_5x_3 + x_2x_3 + x_5 + x_4x_2 + x_3 + x_1x_5 + x_1x_4 + x_1x_6 \\
&\quad + x_1x_2 \\
h_{\alpha^{18}} &= x_6x_4 + x_6 + x_3x_6 + x_6x_2 + x_2 + x_2x_5 + x_4 + x_5 + x_4x_2 + x_1x_5 + x_1x_3 \\
&\quad + x_1x_4 \\
h_{\alpha^{19}} &= x_6x_4 + x_6 + x_3x_6 + x_4 + x_2x_5 + x_5x_3 + x_3 + x_5 + x_1x_3 + x_1x_4 + x_1x_6 \\
&\quad + x_1x_2 \\
h_{\alpha^{20}} &= x_6x_5 + x_6x_4 + x_6 + x_3 + x_2x_5 + x_5x_3 + x_5 + x_4x_2 + x_4 + x_2 + x_1x_5 \\
&\quad + x_1x_3 + x_1x_2 \\
h_{\alpha^{21}} &= x_6x_5 + x_3x_6 + x_6x_4 + x_6 + x_2 + x_3 + x_4 + x_4x_5 + x_5 + x_4x_2 + x_5x_3 \\
&\quad + x_1x_4 + x_1x_6 + x_1x_2 \\
h_{\alpha^{22}} &= x_6x_5 + x_3x_6 + x_6x_4 + x_6 + x_3x_4 + x_4 + x_2x_5 + x_4x_5 + x_5 + x_5x_3 + x_4x_2 \\
&\quad + x_3 + x_2 + x_1x_5 + x_1x_3 \\
h_{\alpha^{23}} &= x_6x_5 + x_3x_6 + x_6x_2 + x_6 + x_2 + x_2x_3 + x_4 + x_3 + x_3x_4 + x_5x_3 + x_5 \\
&\quad + x_2x_5 + x_4x_5 + x_4x_2 + x_1x_4 + x_1x_2 \\
h_{\alpha^{24}} &= x_6x_5 + x_3x_6 + x_6 + x_4x_5 + x_4x_2 + x_2x_5 + x_2x_3 + x_3 + x_4 + x_5 + x_2 \\
&\quad + x_3x_4 + x_1x_3 \\
h_{\alpha^{25}} &= x_3x_6 + x_6 + x_3 + x_5 + x_2 + x_2x_3 + x_2x_5 + x_4x_5 + x_3x_4 + x_4 + x_1x_2 \\
h_{\alpha^{26}} &= x_3 + x_2x_3 + x_3x_4 + x_2x_5 + x_4 + x_2 + x_5 + x_1x_6 \\
h_{\alpha^{27}} &= x_6x_4 + x_2 + x_2x_3 + x_3 + x_4 + x_1x_5 \\
h_{\alpha^{28}} &= x_6x_2 + x_6x_5 + x_5x_3 + x_3 + x_2 + x_1x_4 \\
h_{\alpha^{29}} &= x_6 + x_6x_5 + x_4x_5 + x_2 + x_4x_2 + x_1x_3 + x_1x_6 \\
h_{\alpha^{30}} &= x_6x_4 + x_3x_6 + x_5 + x_4x_5 + x_3x_4 + x_1x_5 + x_1x_2
\end{aligned}
$$

There are two alternative ways to consider the construction of the set of Hadamard matrices from either of these sets of bent functions. The first is to take a $2^m \times 2^m$ Hadamard matrix in Sylvester form and multiply the ith column of the matrix by the value of $\tilde{h}_\gamma(x_1, x_2, \ldots, x_m)$ corresponding to that column. For each distinct choice of γ a new Hadamard matrix is obtained and the inner product of rows of distinct Hadamard matrices has magnitude $2^{m/2}$. The second way is to consider the first order Reed-Muller code S generated by the Boolean functions $x_1, x_2, \ldots, x_m$. The matrix with the codewords of this code as rows (in a certain order) is then the same as the above Hadamard matrix if each element is mapped to $+1$ or -1 by the mapping $g : x \mapsto (-1)^x$. The other Hadamard matrices are obtained in the same way from the translates $S + \hat{\mathbf{h}}_\gamma$ for $\gamma \in \mathrm{GF}(2^{m-1})$.

Theorem 1 *[29]. The above construction gives 2^{m-1} Hadamard matrices with the inner product of rows of distinct matrices having magnitude $2^{m/2}$.*

5.2 Construction using almost bent functions in the case of m odd.

Choose an integer r with m, r relatively prime. Let $x = \sum_{i=1}^{m} x_i\xi_i$ be an expression for an element x of $GF(2^m)$ in terms of a basis $\xi_1, ..., \xi_m$. Let α be a primitive element of $GF(2^m)$. In the examples below $\xi_1 = 1$, $\xi_2 = \alpha$, ..., $\xi_m = \alpha^{m-1}$. Then an *almost bent* function is defined in [25] for any $\gamma \in GF(2^m)$ by

$$h_\gamma(x_1, x_2, ..., x_m) = \mathrm{Tr}_m(\gamma(x_1\xi_1 + x_2\xi_2 + x_3\xi_3 + ... + x_m\xi_m)^{1+2^r}) \qquad (5.1)$$

In fact if the nonzero vectors are arranged as the representation (in terms of the basis) of powers of a primitive element of the field, the expression (5.1) gives a decimated maximal length sequence, as used in the construction of Gold codes. The extra position $(x_1, x_2, \ldots, x_m) = (0, 0, \ldots, 0)$ is used to extend the Gold code. In a similar way, the functions given by $\mathrm{Tr}_m(\gamma(x_1\xi_1 + x_2\xi_2 + x_3\xi_3 + ... + x_m\xi_m))$ would generate a maximal length sequence code if this order of positions is used.

If we take $m = 5$, $r = 2$, and construct the Galois field $GF(2^m)$ using the irreducible polynomial $\alpha^5 + \alpha^2 + 1 = 0$, the almost bent functions can be computed (using the same Maple simplifications as previously) as follows:

$$\begin{aligned}
h_0 &= 0\\
h_1 &= x_3 + x_2 + x_3x_4 + x_5 + x_2x_5 + x_2x_4 + x_1\\
h_\alpha &= x_3 + x_2x_1 + x_3x_2 + x_2 + x_5 + x_3x_5 + x_3x_4 + x_1x_4\\
h_{\alpha^2} &= x_2x_5 + x_3x_4 + x_4 + x_3x_2 + x_5 + x_3 + x_1x_3 + x_3x_5 + x_1x_4 + x_2x_4\\
h_{\alpha^3} &= x_3 + x_1 + x_1x_4 + x_2x_4 + x_3x_5 + x_4x_5 + x_4 + x_3x_4\\
h_{\alpha^4} &= x_2 + x_4x_5 + x_3x_5 + x_3x_4 + x_1x_5 + x_2x_1 + x_5\\
h_{\alpha^5} &= x_1x_4 + x_1 + x_1x_3 + x_3x_5 + x_4 + x_3x_2 + x_2\\
h_{\alpha^6} &= x_1 + x_2x_1 + x_4x_5 + x_4 + x_2 + x_3x_2 + x_2x_4 + x_5\\
h_{\alpha^7} &= x_3x_2 + x_1x_4 + x_1x_3 + x_2x_1 + x_1x_5 + x_2x_4 + x_3 + x_4 + x_4x_5 + x_2 + x_2x_5\\
h_{\alpha^8} &= x_4x_5 + x_2 + x_3x_2 + x_3x_4 + x_1x_3 + x_3 + x_2x_4\\
h_{\alpha^9} &= x_4 + x_1 + x_1x_5 + x_3x_2 + x_3x_5 + x_3x_4 + x_2x_4\\
h_{\alpha^{10}} &= x_3x_5 + x_1x_5 + x_2x_4 + x_2x_5 + x_3 + x_4x_5 + x_2x_1 + x_1\\
h_{\alpha^{11}} &= x_5 + x_4 + x_2x_1 + x_3x_4 + x_3 + x_1 + x_1x_3\\
h_{\alpha^{12}} &= x_4x_5 + x_2x_1 + x_3x_5 + x_1x_4 + x_3 + x_2 + x_3x_4 + x_2x_5 + x_1x_3 + x_1\\
h_{\alpha^{13}} &= x_1 + x_3x_5 + x_3x_4 + x_2x_1 + x_2 + x_2x_5 + x_3x_2 + x_1x_3 + x_1x_5
\end{aligned}$$

$$
\begin{aligned}
h_{\alpha^{14}} &= x_5 + x_1x_5 + x_3 + x_3x_5 + x_2x_1 + x_2x_4 + x_3x_2 + x_1x_3 \\
h_{\alpha^{15}} &= x_1x_3 + x_1x_5 + x_3x_4 + x_2 + x_2x_4 + x_1x_4 \\
h_{\alpha^{16}} &= x_3 + x_2 + x_4 + x_3x_2 + x_2x_5 + x_3x_5 + x_5 + x_1x_5 \\
h_{\alpha^{17}} &= x_1x_4 + x_2x_5 + x_4x_5 + x_5 + x_1x_5 + x_2 + x_1 + x_3x_2 + x_2x_4 + x_3x_4 \\
h_{\alpha^{18}} &= x_2x_5 + x_2x_1 + x_1x_4 + x_2x_4 + x_1 + x_3x_2 + x_3x_5 \\
h_{\alpha^{19}} &= x_1x_3 + x_2x_1 + x_2 + x_2x_4 + x_2x_5 + x_4 \\
h_{\alpha^{20}} &= x_2x_5 + x_1 + x_4x_5 + x_5 + x_1x_3 + x_3x_2 \\
h_{\alpha^{21}} &= x_1x_4 + x_1 + x_4 + x_2x_1 + x_2x_4 + x_1x_5 + x_2 + x_5 + x_3 \\
h_{\alpha^{22}} &= x_5 + x_1 + x_3x_4 + x_2x_1 + x_1x_4 + x_1x_3 + x_1x_5 + x_3x_2 + x_4x_5 + x_4 \\
h_{\alpha^{23}} &= x_2x_4 + x_1x_3 + x_5 + x_1x_4 + x_4x_5 + x_2x_1 + x_3x_5 \\
h_{\alpha^{24}} &= x_2x_5 + x_3 + x_1x_3 + x_5 + x_1x_4 + x_1x_5 + x_1 \\
h_{\alpha^{25}} &= x_1x_5 + x_1x_4 + x_4 + x_2x_5 + x_3x_4 + x_2x_1 \\
h_{\alpha^{26}} &= x_1x_3 + x_1x_5 + x_3 + x_1 + x_2 + x_3x_5 + x_4 + x_4x_5 \\
h_{\alpha^{27}} &= x_3 + x_3x_2 + x_2x_5 + x_3x_4 + x_2x_1 + x_4 + x_4x_5 \\
h_{\alpha^{28}} &= x_2x_5 + x_2x_4 + x_5 + x_1x_5 + x_4 + x_3x_4 + x_3x_5 + x_1x_3 + x_4x_5 \\
h_{\alpha^{29}} &= x_2 + x_2x_5 + x_5 + x_1x_4 + x_4x_5 + x_3x_5 + x_4 \\
h_{\alpha^{30}} &= x_3 + x_1x_4 + x_1x_5 + x_3x_2 + x_4x_5
\end{aligned}
$$

Again there are two alternative ways to consider the construction of the set of Hadamard matrices from this set of almost bent functions. The first is to take a $2^m \times 2^m$ Hadamard matrix in Sylvester form and multiply the ith column of the matrix by the value of $\tilde{h}_\gamma(x_1, x_2, \ldots, x_m)$ corresponding to that column. For each distinct choice of γ a new Hadamard matrix is obtained and the inner product of rows of distinct Hadamard matrices has magnitude at most $2^{(m+1)/2}$. The second way is to consider the first order Reed-Muller code S generated by the Boolean functions $x_1, x_2, \ldots, x_m$. The matrix with the codewords of this code as rows (in a certain order) is then the same as the above Hadamard matrix if each element is mapped to $+1$ or -1 by the mapping $g : x \mapsto (-1)^x$. The other Hadamard matrices are obtained in the same way from the cosets $S + \hat{\mathbf{h}}_\gamma$ for $\gamma \in \mathrm{GF}(2^m)$.

Theorem 2 *[29]. The above construction gives 2^m Hadamard matrices with the inner product of rows of distinct matrices having magnitude at most $2^{(m+1)/2}$.*

5.3 A potential security concern and partitions of Gold codes

Consider the case where all of the information required to evolve the code of a single user becomes known to a third party. Then the Golay pairs and internal

padding of all users in each generation are known to the third party. If each user is only given information on an individual Hadamard matrix row and how it evolves, then full information on the codewords of other users is not available to the third party. Consider, however, the case where the third party has learned all Hadamard matrix rows used by all users during evolution of the system. If the number of partitions into Hadamard matrices is sufficiently large, then knowledge of one user's Hadamard row gives no information on any other user's Hadamard row. However, if the partition of all Hadamard matrix rows into Hadamard matrices is unique, then this third party knows the Golay pair, internal padding vector and Hadamard matrix row for all users in the current generation. The concern remains if the partition can occur in only a small number of ways.

For Kerdock codes the definition of bent functions implies that the only zero inner products are those between rows of the matrices arising from the same bent function. Thus the partition into Hadamard matrices given by the bent functions is unique and the security concern is a very real one.

For m odd the almost bent function allows zero inner products between rows of distinct matrices. In fact the original partition can be modified to create many different Hadamard matrix partitions of the set of rows of the original matrices. Thus, in contrast to the case of m even, the security concern does not arise for m odd. In the remained of this section the number of partitions of extended Gold codes into Hadamard matrices in the case of m odd is determined. A detailed account of this work can also be found in [20].

Here it is particularly convenient to consider the correlation of vectors as *number of agreements − number of disagreements* taken over all positions, as this gives a uniform definition for correlation of codewords of $\{0, 1\}$ codes and inner products of rows of $\{+1, -1\}$ matrices.

5.3.1 Properties of the almost bent functions for m odd.

It should be noted from equation (5.1) that (for odd m) the almost bent functions are linear functions on γ, i.e. $h_{\gamma_1} + h_{\gamma_2} = h_{\gamma_1+\gamma_2}$. For a Boolean variable x_i the notation $x_i + 1$ will be used for $x_i + 1 \bmod 2$. Then for the functions defined in (5.1) (with $r = 2$) the following properties are easily checked:

$$
\begin{aligned}
h_1(x_1,x_2,x_3,x_4,x_5) &= h_1(x_1+1,x_2,x_3,x_4,x_5)+1\\
h_\alpha(x_1,x_2,x_3,x_4,x_5) &= h_\alpha(x_1,x_2+1,x_3,x_4+1,x_5)+1\\
h_{\alpha^2}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^2}(x_1,x_2+1,x_3+1,x_4+1,x_5)+1\\
h_{\alpha^3}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^3}(x_1+1,x_2+1,x_3,x_4,x_5)+1\\
h_{\alpha^4}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^4}(x_1,x_2+1,x_3+1,x_4+1,x_5+1)+1\\
h_{\alpha^5}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^5}(x_1,x_2+1,x_3,x_4,x_5+1)+1\\
h_{\alpha^6}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^6}(x_1+1,x_2,x_3+1,x_4,x_5)+1\\
h_{\alpha^7}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^7}(x_1+1,x_2+1,x_3+1,x_4,x_5)+1\\
h_{\alpha^8}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^8}(x_1+1,x_2+1,x_3,x_4,x_5+1)+1\\
h_{\alpha^9}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^9}(x_1+1,x_2+1,x_3+1,x_4+1,x_5)+1\\
h_{\alpha^{10}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{10}}(x_1+1,x_2,x_3,x_4+1,x_5)+1\\
h_{\alpha^{11}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{11}}(x_1,x_2,x_3,x_4,x_5+1)+1\\
h_{\alpha^{12}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{12}}(x_1+1,x_2,x_3,x_4,x_5+1)+1\\
h_{\alpha^{13}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{13}}(x_1+1,x_2+1,x_3,x_4+1,x_5+1)+1\\
h_{\alpha^{14}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{14}}(x_1+1,x_2,x_3+1,x_4,x_5+1)+1\\
h_{\alpha^{15}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{15}}(x_1,x_2+1,x_3+1,x_4,x_5+1)+1\\
h_{\alpha^{16}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{16}}(x_1,x_2,x_3,x_4+1,x_5)+1\\
h_{\alpha^{17}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{17}}(x_1,x_2+1,x_3,x_4+1,x_5+1)+1\\
h_{\alpha^{18}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{18}}(x_1+1,x_2+1,x_3+1,x_4+1,x_5+1)+1\\
h_{\alpha^{19}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{19}}(x_1,x_2,x_3,x_4+1,x_5+1)+1\\
h_{\alpha^{20}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{20}}(x_1+1,x_2+1,x_3,x_4+1,x_5)+1\\
h_{\alpha^{21}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{21}}(x_1,x_2,x_3+1,x_4,x_5)+1\\
h_{\alpha^{22}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{22}}(x_1+1,x_2,x_3+1,x_4+1,x_5)+1\\
h_{\alpha^{23}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{23}}(x_1+1,x_2,x_3+1,x_4+1,x_5+1)+1\\
h_{\alpha^{24}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{24}}(x_1,x_2,x_3+1,x_4+1,x_5)+1\\
h_{\alpha^{25}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{25}}(x_1+1,x_2+1,x_3+1,x_4,x_5+1)+1\\
h_{\alpha^{26}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{26}}(x_1,x_2+1,x_3,x_4,x_5)+1\\
h_{\alpha^{27}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{27}}(x_1,x_2,x_3+1,x_4,x_5+1)+1\\
h_{\alpha^{28}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{28}}(x_1,x_2,x_3+1,x_4+1,x_5+1)+1\\
h_{\alpha^{29}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{29}}(x_1,x_2+1,x_3+1,x_4,x_5)+1\\
h_{\alpha^{30}}(x_1,x_2,x_3,x_4,x_5) &= h_{\alpha^{30}}(x_1+1,x_2,x_3,x_4+1,x_5+1)+1
\end{aligned}
$$

Observation of the properties in this example leads to the following theorem:

Theorem 3 *Let the expression:*

$$h_\gamma(\mathbf{x}) = h_\gamma(x_1, x_2, ..., x_m) = \mathrm{Tr}_m(\gamma(x_1\xi_1 + x_2\xi_2 + x_3\xi_3 + ... + x_m\xi_m)^{1+2^r})$$

be used to generate the almost bent functions. If $r = 2$ and $m \not\equiv 0 \bmod 4$ then for each nonzero γ in $GF(2^m)$ there exists a nonzero binary vector Δ such that $h_\gamma(\mathbf{x}+\Delta) = h_\gamma(\mathbf{x}) + \delta'$ with $\delta' \in \{0, 1\}$. If m is odd then $\delta' = 1$ and each nonzero value of Δ occurs for a distinct nonzero value of γ.

Proof Let $\Delta = (\delta_1, \delta_2, \ldots, \delta_m)$, $x = x_1\xi_1 + x_2\xi_2 + x_3\xi_3 + ... + x_m\xi_m \in GF(2^m)$ and $\Lambda = \delta_1\xi_1 + \delta_2\xi_2 + \delta_3\xi_3 + ... + \delta_m\xi_m \in GF(2^m)$, so $\mathbf{x}$ and Δ are representations of x and Λ in terms of the chosen basis. In a field of characteristic 2, $(x+y)^p = x^p + y^p$ when p is a power of 2, so $(x + y)^{1+2^r} = (x + y)(x^{2^r} + y^{2^r})$. Thus

$$h_\gamma(\mathbf{x} + \Delta) = \mathrm{Tr}_m(\gamma(x + \Lambda)^{1+2^r}) = \mathrm{Tr}_m(\gamma(x^{1+2^r} + x^{2^r}\Lambda + x\Lambda^{2^r} + \Lambda^{1+2^r})$$

$$= \mathrm{Tr}_m(\gamma x^{1+2^r}) + \mathrm{Tr}_m(\gamma x^{2^r}\Lambda) + \mathrm{Tr}_m(\gamma x\Lambda^{2^r}) + \mathrm{Tr}_m(\gamma\Lambda^{1+2^r})$$

Recalling that $\mathrm{Tr}_m(\beta^{2^r}) = \mathrm{Tr}_m(\beta)$ for all $\beta \in \mathrm{GF}(2^m)$, it can be seen that the middle terms disappear if $\gamma x^{2^r}\Lambda = (\gamma x\Lambda^{2^r})^{2^r}$, i.e $1 = \gamma^{2^r-1}\Lambda^{2^{2r}-1}$. Then

$$h_\gamma(\mathbf{x} + \Delta) = h_\gamma(\mathbf{x}) + \delta' \text{ with } \delta' \in \{0, 1\}.$$

For $r = 2$ and $m \not\equiv 0 \bmod 4$, $2^m \equiv 2$, 3, or 4 mod 5 and there is a solution of $\gamma^3\Lambda^{15} = 1$ for each nonzero value of γ. There are three cases:

1. $2^m = 5t + 2$, $\gamma^{5t+1} = 1$ and $\Lambda = \gamma^{\,t}$ is a solution.
2. $2^m = 5t + 3$, $\gamma^{15t+6} = 1$ and $\Lambda = \gamma^{3t+1}$ is a solution.
3. $2^m = 5t + 4$, $\gamma^{10t+6} = 1$ and $\Lambda = \gamma^{2t+1}$ is a solution.

If m is odd then $2^m - 1 \not\equiv 0 \bmod 3$. Let $\Lambda = \alpha^j$ and $\gamma^3 = \Lambda^{-15} = \alpha^{i(2^m-1)} \times \alpha^{-15j}$. Thus $i = 0$ or i is divisible by 3 and the solution $\gamma = \Lambda^{-5}$ is unique. Thus each distinct vector Δ corresponds to a unique almost bent function h_γ. Also, $\delta' = \mathrm{Tr}_m(\gamma\Lambda^5) = \mathrm{Tr}_m(1)$. For m odd $\mathrm{Tr}_m(1) = 1$ so $\delta' = 1$.

There is a possibility that conditions other than $1 = \gamma^{2^r-1}\Lambda^{2^{2r}-1}$ could lead to different sets of vectors Δ with this property. However, in this chapter we are concerned with lower bounds, and so this possibility need not be considered.

5.3.2 Partitions into Hadamard matrices

The partition into Hadamard matrices is obtained from a subcode of a first order Reed-Muller code and its cosets (obtained by adding the codewords corresponding to the almost bent functions) in the Gold code. The partition of the extended $\{0,1\}$ Gold code obtained can then be mapped to a partition into $\{+1,-1\}$ Hadamard matrices by the mapping $g : x \mapsto (-1)^x$. Because of the definition of the inner product of rows of the matrix as *number of agreements − number of disagreements* taken over all positions, it is sufficient to work with the $\{0,1\}$ codewords.

Let S be the subcode of the first order Reed-Muller code, generated by the linear Boolean functions $\phi_1(x_1, x_2, \ldots, x_m) = x_1, \phi_2(x_1, x_2, \ldots, x_m) = x_2, \ldots, \phi_m(x_1, x_2, \ldots, x_m) = x_m$ evaluated (using the vectors indexing the positions) to give codewords. Let $\hat{h}_\gamma$ be the codeword corresponding to the almost bent function h_γ, evaluated in the same way. The union of cosets $\cup_{\gamma \in GF(2^m)}(S + \hat{h}_\gamma)$ of the $\{0,1\}$ code then corresponds to the initial partition into $\{+1,-1\}$ Hadamard matrices.

Now consider the two cosets $K' = S + \hat{h}_{\gamma_1}$ and $K'' = S + \hat{h}_{\gamma_2}$. From equation (5.1) and the linearity of the trace function, the bent functions are linear on γ and $h_{\gamma_1} + h_{\gamma_2} = h_{\gamma_3}$ where $\gamma_1 + \gamma_2 = \gamma_3$. Consider the vector $\Delta = (\delta_1, \delta_2, \ldots, \delta_m)$ corresponding to h_{γ_3}. Let K_1 be the subcode of S generated by the linear Boolean functions $\sum_{i=1}^m \mu_i x_i$ which satisfy $\sum_{i=1}^m \delta_i \mu_i = 0$. Also, let K_2 be the coset of K_1 for which the functions satisfy $\sum_{i=1}^m \delta_i \mu_i = 1$. Then $S = K_1 \cup K_2$, $K' = (K_1 \cup K_2) + \hat{h}_{\gamma_1}$, $K'' = (K_1 \cup K_2) + \hat{h}_{\gamma_2}$ and the initial partition can be written $\cup_{\gamma \in GF(2^m)}((K_1 \cup K_2) + \hat{h}_\gamma)$. Define also $S' = (K_1 \cup K_2) + \hat{h}_{\gamma_3}$.

Let $\mathbf{c}_1 \in S$, $\mathbf{c}_2 \in S'$, $\mathbf{c}_3 = \mathbf{c}_1 + \hat{h}_{\gamma_1} \in K'$, $\mathbf{c}_4 = \mathbf{c}_2 + \hat{h}_{\gamma_1} \in K''$. Denote by $A(\mathbf{c}_1, \mathbf{c}_2)$ the number of positions for which $\mathbf{c}_1$ and $\mathbf{c}_2$ agree and by $D(\mathbf{c}_1, \mathbf{c}_2)$ the number of positions for which $\mathbf{c}_1$ and $\mathbf{c}_2$ disagree. Then as $\mathbf{c}_3 = \mathbf{c}_1 + \hat{h}_{\gamma_1} \in K'$, $\mathbf{c}_4 = \mathbf{c}_2 + \hat{h}_{\gamma_1} \in K''$, $\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \mathbf{c}_4$ satisfy $A(\mathbf{c}_1, \mathbf{c}_2) - D(\mathbf{c}_1, \mathbf{c}_2) = 0$ if and only if $A(\mathbf{c}_3, \mathbf{c}_4) - D(\mathbf{c}_3, \mathbf{c}_4) = 0$. Thus if it can be shown that S, S' lead to two different (sets of codewords which map to) Hadamard matrices then K', K'' lead to two different (sets of codewords which map to) Hadamard matrices. For each vector $\boldsymbol{\Delta}$ create a partition $\{P, \bar{P}\}$ of the positions such that $(x_1, x_2, \ldots, x_m) \in P$ if and only if $(x_1 + \delta_1, x_2 + \delta_2, \ldots, x_m + \delta_m) \in \bar{P}$.

Starting from S and S' create two sets of codewords $L = K_1 \cup (K_1 + \hat{h}_{\gamma_3})$ and $L' = K_2 \cup (K_2 + \hat{h}_{\gamma_3})$. This will be referred to as a *switching* operation and is illustrated in figure 5.1. Consider the first of these sets. For two codewords $\mathbf{c}$ and $\mathbf{c}'$ both in K_1 or both in $(K_1 + \hat{h}_{\gamma_3})$, $A(\mathbf{c}, \mathbf{c}') - D(\mathbf{c}, \mathbf{c}') = 0$ follows from the

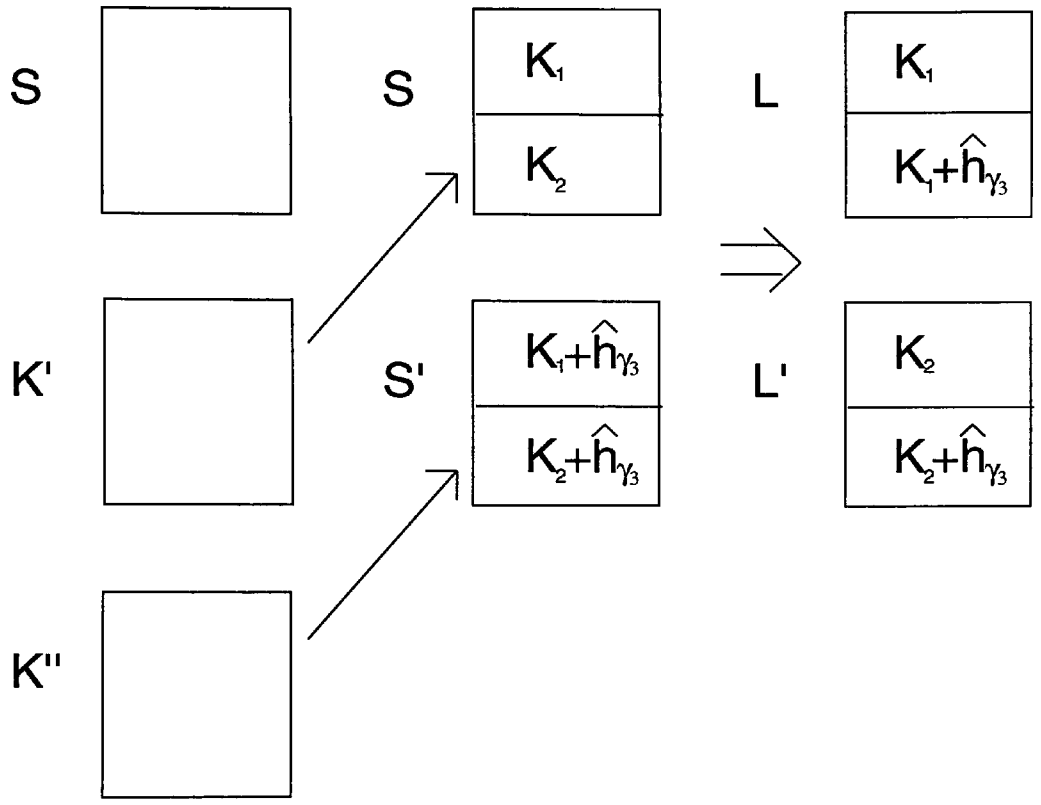

Figure 5.1: *Illustration of the switching operation.*

definition of S. Thus it is only necessary to consider $\mathbf{c} \in K_1$ and $\mathbf{c}' \in K_1 + \hat{h}_{\gamma_3}$. If $\mathbf{c} \in K_1$ corresponds to the function $\sum_{i=1}^{m} \mu_i x_i$ then $\sum_{i=1}^{m} \mu_i(x_i + \delta_i)$ has the same value as $\sum_{i=1}^{m} \mu_i x_i$. For $\mathbf{c}'$, $\sum_{i=1}^{m} \mu_i'(x_i + \delta_i) + h_{\gamma_3}(\mathbf{x} + \Delta) = \sum_{i=1}^{m} \mu_i' x_i + h_{\gamma_3}(\mathbf{x}) + 1$ follows from theorem 3. Thus an agreement of these two codewords $\mathbf{c}, \mathbf{c}'$ on a position within the set P gives a disagreement on the corresponding position of $\bar{P}$ and a disagreement on a position in P gives an agreement on the corresponding position of $\bar{P}$. Considering all the positions of $P \cup \bar{P}$ it follows that $A(\mathbf{c}, \mathbf{c}') - D(\mathbf{c}, \mathbf{c}') = 0$. A similar argument deals with the case $L' = K_2 \cup (K_2 + \hat{h}_{\gamma_3})$. Thus L and L' both map to Hadamard matrices and in general $(K_1 + \hat{h}_{\gamma_1}) \cup (K_1 + \hat{h}_{\gamma_2})$ and $(K_2 + \hat{h}_{\gamma_1}) \cup (K_2 + \hat{h}_{\gamma_2})$ both map to Hadamard matrices.

Construct a complete graph G with 2^m vertices corresponding to cosets $(K_1 \cup K_2) + \hat{h}_\gamma$. Consider each (not necessarily perfect) matching of G, consisting of a set of edges for which no pair are incident with the same vertex. For such a matching, with between 1 and 2^{m-1} edges, a new partition into Hadamard

matrices is obtained as indicated above. Each edge of the matching gives a pair of almost bent functions and from their sum an almost bent function corresponding to the edge is obtained. This almost bent function gives unique values of Δ and δ', which show how to split the cosets into halves and pair the halves. The number of partitions obtained by the above method is the number of matchings in G. There are

$$(2u-1)!! = (2u-1)(2u-3)(2u-5)\ldots 5.3.1 = \frac{(2u-1)!}{(u-1)!2^{u-1}}$$

perfect matchings in a complete graph K_{2u} with $2u$ vertices.

In general, for a complete graph with n vertices containing a matching with u edges, we can select the vertices incident with edges of the matching in $\binom{2^m}{2u}$ ways. For the complete graph K_{2u} induced by these vertices, the perfect matching can be selected in $\frac{(2u-1)!}{(u-1)!2^{u-1}}$ ways. Thus the total number of partitions (including the original partition) obtained by the above method is

$$1 + \sum_{u=1}^{2^{m-1}} \binom{2^m}{2u} \frac{(2u-1)!}{(u-1)!2^{u-1}}.$$

For $m = 5$ this number is 22481059424730751232. Thus we have the following theorem.

Theorem 4 *If m is odd and $r = 2$, the number of distinct partitions into Hadamard matrices obtained using the initial partition into Hadamard matrices given by the almost bent functions is at least*

$$1 + \sum_{u=1}^{2^{m-1}} \binom{2^m}{2u} \frac{(2u-1)!}{(u-1)!2^{u-1}}.$$

Clique search has revealed another type of partition. Each Hadamard matrix in this type consists of exactly one row of each Hadamard matrix of the original partition. The way that such partitions arise will now be described.

Let $r = 2$ and consider first the case $m = 5$. For each $\gamma \in GF(2^m)$ the functions

$$H_\gamma(x_1, x_2, ..., x_m) = \mathrm{Tr}_m((\alpha^j\gamma + \alpha^{2j}\gamma^8)x) + h_\gamma(x_1, x_2, ..., x_m) \qquad (5.2)$$

($j \in \{0, \ldots, 2^m - 2\}$), together with the function $H_\gamma(x_1, x_2, ..., x_m) = h_\gamma(x_1, x_2, ..., x_m)$, form a set of 2^m functions. For fixed j these functions $H_\gamma(x_1, x_2, ..., x_m)$ are linear on γ. By choosing distinct values of γ, one codeword $\hat{H}_\gamma(x_1, x_2, ..., x_m)$ is chosen from each coset of S, as the first term in (5.2) gives first

order Boolean functions. Linearity follows from $\gamma_1^{2^t}+\gamma_2^{2^t}=(\gamma_1+\gamma_2)^{2^t}$ and the linearity of the trace function [14]. If $H_\gamma(x_1,x_2,...,x_m)=h_\gamma(x_1,x_2,...,x_m)$ it has already been shown that for $\gamma\Lambda^5=1$, $H_\gamma(\mathbf{x}+\mathbf{\Delta})=H_\gamma(\mathbf{x})+1$. If $H_\gamma(x_1,x_2,...,x_m)$ is as given in (5.2) then $H_\gamma(\mathbf{x}+\mathbf{\Delta})=H_\gamma(\mathbf{x})+\mathrm{Tr}_m((\alpha^j\gamma+\alpha^{2j}\gamma^8)\Lambda)+1$. But for $m=5$ the general condition $\gamma^6\Lambda^{30}=1$ implies $\Lambda=\gamma^6$ and so $\mathrm{Tr}_m((\alpha^j\gamma+\alpha^{2j}\gamma^8)\Lambda)=\mathrm{Tr}_m((\alpha^j\gamma)\Lambda)+\mathrm{Tr}_m((\alpha^{2j}\gamma^8)\Lambda)=\mathrm{Tr}_m(\alpha^j\gamma^7)+\mathrm{Tr}_m(\alpha^{2j}\gamma^{14})=0$ as $\mathrm{Tr}_m(\beta)=\mathrm{Tr}_m(\beta^2)$ for all $\beta\in\mathrm{GF}(2^m)$, i.e.

$$\mathrm{Tr}_m((\alpha^j\gamma+\alpha^{2j}\gamma^8)\Lambda)=0.$$

Thus $H_\gamma(\mathbf{x}+\mathbf{\Delta})=H_\gamma(\mathbf{x})+1$ in the general case. The result that $A(\hat{H}_{\gamma_1}(x_1,x_2,...,x_m),\hat{H}_{\gamma_2}(x_1,x_2,...,x_m))-D(\hat{H}_{\gamma_1}(x_1,x_2,...,x_m),\hat{H}_{\gamma_2}(x_1,x_2,...,x_m))$ is zero follows in the same way as previously. Thus the set of codewords $\hat{H}_\gamma(x_1,x_2,...,x_m)$ for $\gamma\in\mathrm{GF}(2^m)$ forms a subspace corresponding to a Hadamard matrix. This subspace and its cosets in the extended Gold code map to a new partition into Hadamard matrices, with exactly one row of each Hadamard matrix chosen from each of the original Hadamard matrices. In the case $m=5$, $r=2$ there are 2^m such subspaces and therefore 2^m such Hadamard matrix partitions. Thus for $m=5$ the number of partitions counted becomes

$$1+\sum_{u=1}^{2^{m-1}}\binom{2^m}{2u}\frac{(2u-1)!}{(u-1)!2^{u-1}}+2^m. \tag{5.3}$$

In the case $m=5$ the 31 values $\{0,1,\ldots,30\}$ for j together with the case given by $H_\gamma(\mathbf{x})=h_\gamma(\mathbf{x})$ explain precisely the 32 cases of the "one row of each Hadamard matrix" type observed by clique search. However, the precise structure may be different for different (odd) values of m. Observing the first term of equation (5.2) and noting the requirement for linearity, pairs of values of $\gamma^{2^i}\Lambda$, $\gamma^{2^k}\Lambda$ are required in the same cyclotomic coset (i.e. so that $\gamma^{2^k}\Lambda=(\gamma^{2^i}\Lambda)^{2^t}$).

The case $m=3$

Here $\gamma^7=\Lambda^7=1$ and so $\gamma\Lambda^5=1$ implies $\Lambda=\gamma^4$. The cyclotomic cosets are:

$$\gamma^l\Lambda\in\{\gamma,\gamma^2,\gamma^4\}\text{ so }\gamma^l\in\{\gamma^4,\gamma^5,1\}$$
$$\gamma^l\Lambda\in\{\gamma^5,\gamma^3,\gamma^6\}\text{ so }\gamma^l\in\{\gamma,\gamma^6,\gamma^2\}.$$

Thus the only possibility which gives the necessary linearity is to use γ^2 and γ for γ^l (corresponding to the second cyclotomic coset). Note that the choice γ and γ^2 gives the same result. If the zero function is included, this gives a set of 2^m functions

$$\mathrm{Tr}_m((\alpha^j\gamma^2+\alpha^{2j}\gamma)x)$$

giving $H_\gamma(x_1,x_2,...,x_m)=h_\gamma(x_1,x_2,...,x_m)$ and

$$H_\gamma(x_1,x_2,...,x_m)\;=\;\mathrm{Tr}_m((\alpha^j\gamma^2+\alpha^{2j}\gamma)x)+h_\gamma(x_1,x_2,...,x_m) \tag{5.4}$$

and the number of partitions obtained is again given by (5.3).

The case $m = 5$ (to check completeness)

Here $\gamma^{31} = \Lambda^{31} = 1$ and so $\gamma\Lambda^5 = 1$ implies $\Lambda = \gamma^6$. The cyclotomic cosets are:

$$\gamma^l\Lambda \in \{\gamma^7, \gamma^{14}, \gamma^{28}, \gamma^{25}, \gamma^{19}\}; \quad \gamma^l \in \{\gamma, \gamma^8, \gamma^{22}, \gamma^{19}, \gamma^{13}\}$$

$$\gamma^l\Lambda \in \{\gamma^8, \gamma^{16}, \gamma, \gamma^2, \gamma^4\}; \quad \gamma^l \in \{\gamma^2, \gamma^{10}, \gamma^{26}, \gamma^{27}, \gamma^{29}\}$$

$$\gamma^l\Lambda \in \{\gamma^9, \gamma^{18}, \gamma^5, \gamma^{10}, \gamma^{20}\}; \quad \gamma^l \in \{\gamma^3, \gamma^{12}, \gamma^{30}, \gamma^4, \gamma^{14}\}$$

$$\gamma^l\Lambda \in \{\gamma^{11}, \gamma^{22}, \gamma^{13}, \gamma^{26}, \gamma^{21}\}; \quad \gamma^l \in \{\gamma^5, \gamma^{16}, \gamma^7, \gamma^{20}, \gamma^{15}\}$$

$$\gamma^l\Lambda \in \{\gamma^{12}, \gamma^{24}, \gamma^{17}, \gamma^3, \gamma^6\}; \quad \gamma^l \in \{\gamma^6, \gamma^{18}, \gamma^{11}, \gamma^{28}, 1\}$$

$$\gamma^l\Lambda \in \{\gamma^{15}, \gamma^{30}, \gamma^{29}, \gamma^{27}, \gamma^{23}\}; \quad \gamma^l \in \{\gamma^9, \gamma^{24}, \gamma^{23}, \gamma^{21}, \gamma^{17}\}$$

It can be seen that the use of γ and γ^8 for γ^l (as in (5.2)) is the only possibility.

The case $m = 7$

Here $\gamma^{127} = \Lambda^{127} = 1$ and so $\gamma\Lambda^5 = 1$ implies $\Lambda = \gamma^{76}$. The cyclotomic cosets are:

$$\gamma^l\Lambda \in \{\gamma^1, \gamma^2, \gamma^4, \gamma^8, \gamma^{16}, \gamma^{32}, \gamma^{64}\}; \quad \gamma^l \in \{\gamma^{52}, \gamma^{53}, \gamma^{55}, \gamma^{59}, \gamma^{67}, \gamma^{83}, \gamma^{115}\}$$

$$\gamma^l\Lambda \in \{\gamma^9, \gamma^{18}, \gamma^{36}, \gamma^{72}, \gamma^{17}, \gamma^{34}, \gamma^{68}\}; \quad \gamma^l \in \{\gamma^{60}, \gamma^{69}, \gamma^{87}, \gamma^{123}, \gamma^{68}, \gamma^{85}, \gamma^{119}\}$$

$$\gamma^l\Lambda \in \{\gamma^{77}, \gamma^{27}, \gamma^{54}, \gamma^{108}, \gamma^{89}, \gamma^{51}, \gamma^{102}\}; \quad \gamma^l \in \{\gamma, \gamma^{78}, \gamma^{105}, \gamma^{32}, \gamma^{13}, \gamma^{102}, \gamma^{26}\}$$

$$\gamma^l\Lambda \in \{\gamma^{78}, \gamma^{29}, \gamma^{58}, \gamma^{116}, \gamma^{105}, \gamma^{83}, \gamma^{39}\}; \quad \gamma^l \in \{\gamma^2, \gamma^{80}, \gamma^{109}, \gamma^{40}, \gamma^{29}, \gamma^7, \gamma^{90}\}$$

$$\gamma^l\Lambda \in \{\gamma^{79}, \gamma^{31}, \gamma^{62}, \gamma^{124}, \gamma^{121}, \gamma^{115}, \gamma^{103}\}; \quad \gamma^l \in \{\gamma^3, \gamma^{82}, \gamma^{113}, \gamma^{48}, \gamma^{45}, \gamma^{39}, \gamma^{27}\}$$

$$\gamma^l\Lambda \in \{\gamma^{80}, \gamma^{33}, \gamma^{66}, \gamma^5, \gamma^{10}, \gamma^{20}, \gamma^{40}\}; \quad \gamma^l \in \{\gamma^4, \gamma^{84}, \gamma^{117}, \gamma^{56}, \gamma^{61}, \gamma^{71}, \gamma^{91}\}$$

$$\gamma^l\Lambda \in \{\gamma^{81}, \gamma^{35}, \gamma^{70}, \gamma^{13}, \gamma^{26}, \gamma^{52}, \gamma^{104}\}; \quad \gamma^l \in \{\gamma^5, \gamma^{86}, \gamma^{121}, \gamma^{64}, \gamma^{77}, \gamma^{103}, \gamma^{28}\}$$

$$\gamma^l\Lambda \in \{\gamma^{82}, \gamma^{37}, \gamma^{74}, \gamma^{21}, \gamma^{42}, \gamma^{84}, \gamma^{41}\}; \quad \gamma^l \in \{\gamma^6, \gamma^{88}, \gamma^{125}, \gamma^{72}, \gamma^{93}, \gamma^8, \gamma^{92}\}$$

$$\gamma^l\Lambda \in \{\gamma^{85}, \gamma^{43}, \gamma^{86}, \gamma^{45}, \gamma^{90}, \gamma^{53}, \gamma^{106}\}; \quad \gamma^l \in \{\gamma^9, \gamma^{94}, \gamma^{10}, \gamma^{96}, \gamma^{14}, \gamma^{104}, \gamma^{30}\}$$

$$\gamma^l\Lambda \in \{\gamma^{87}, \gamma^{47}, \gamma^{94}, \gamma^{61}, \gamma^{122}, \gamma^{117}, \gamma^{107}\}; \quad \gamma^l \in \{\gamma^{11}, \gamma^{98}, \gamma^{18}, \gamma^{112}, \gamma^{46}, \gamma^{41}, \gamma^{31}\}$$

$$\gamma^l\Lambda \in \{\gamma^{88}, \gamma^{49}, \gamma^{98}, \gamma^{69}, \gamma^{11}, \gamma^{22}, \gamma^{44}\}; \quad \gamma^l \in \{\gamma^{12}, \gamma^{100}, \gamma^{22}, \gamma^{120}, \gamma^{62}, \gamma^{73}, \gamma^{95}\}$$

$$\gamma^l\Lambda \in \{\gamma^{91}, \gamma^{55}, \gamma^{110}, \gamma^{93}, \gamma^{59}, \gamma^{118}, \gamma^{109}\}; \quad \gamma^l \in \{\gamma^{15}, \gamma^{106}, \gamma^{34}, \gamma^{17}, \gamma^{110}, \gamma^{42}, \gamma^{33}\}$$

$$\gamma^l\Lambda \in \{\gamma^{95}, \gamma^{63}, \gamma^{126}, \gamma^{125}, \gamma^{123}, \gamma^{119}, \gamma^{111}\}; \quad \gamma^l \in \{\gamma^{19}, \gamma^{114}, \gamma^{50}, \gamma^{49}, \gamma^{47}, \gamma^{43}, \gamma^{35}\}$$

$$\gamma^l\Lambda \in \{\gamma^{96}, \gamma^{65}, \gamma^3, \gamma^6, \gamma^{12}, \gamma^{24}, \gamma^{48}\}; \quad \gamma^l \in \{\gamma^{20}, \gamma^{116}, \gamma^{54}, \gamma^{57}, \gamma^{63}, \gamma^{75}, \gamma^{99}\}$$

$$\gamma^l\Lambda \in \{\gamma^{97}, \gamma^{67}, \gamma^7, \gamma^{14}, \gamma^{28}, \gamma^{56}, \gamma^{112}\}; \quad \gamma^l \in \{\gamma^{21}, \gamma^{118}, \gamma^{58}, \gamma^{65}, \gamma^{79}, \gamma^{107}, \gamma^{36}\}$$

$$\gamma^l\Lambda \in \{\gamma^{99}, \gamma^{71}, \gamma^{15}, \gamma^{30}, \gamma^{60}, \gamma^{120}, \gamma^{113}\}; \quad \gamma^l \in \{\gamma^{23}, \gamma^{122}, \gamma^{66}, \gamma^{81}, \gamma^{111}, \gamma^{44}, \gamma^{37}\}$$

$$\gamma^l\Lambda \in \{\gamma^{100}, \gamma^{73}, \gamma^{19}, \gamma^{38}, \gamma^{76}, \gamma^{25}, \gamma^{50}\}; \quad \gamma^l \in \{\gamma^{24}, \gamma^{124}, \gamma^{70}, \gamma^{89}, 1, \gamma^{76}, \gamma^{101}\}$$

$$\gamma^l\Lambda \in \{\gamma^{101}, \gamma^{75}, \gamma^{23}, \gamma^{46}, \gamma^{92}, \gamma^{57}, \gamma^{114}\}; \quad \gamma^l \in \{\gamma^{25}, \gamma^{126}, \gamma^{74}, \gamma^{97}, \gamma^{16}, \gamma^{108}, \gamma^{38}\}$$

Thus the only possibility is to use γ and γ^{32} for γ^l (corresponding to the third cyclotomic coset). This gives a set of 2^m functions for each γ, consisting of $h_\gamma(x_1, x_2, ..., x_m)$ and (for the $2^m - 1$ choices of j)

$$H_\gamma(x_1, x_2, ..., x_m) = \mathrm{Tr}_m((\alpha^j\gamma + \alpha^{8j}\gamma^{32})x) + h_\gamma(x_1, x_2, ..., x_m) \quad (5.5)$$

and again the number of partitions is counted by (5.2).

It is unclear whether there will always be an appropriate choice of γ^{2^i} and γ^{2^k} for odd $m \geq 9$, or whether there may be more than one choice.

Of course there could be other partitions into Hadamard matrices obtained by less regular methods, but clique searches have not revealed any.

Theorem 5 *If $r = 2$ and m is 3,5 or 7 then the number of distinct partitions into Hadamard matrices obtained using the initial partition into Hadamard matrices given by the almost bent functions is at least*

$$1 + \sum_{u=1}^{2^{m-1}} \binom{2^m}{2u} \frac{(2u-1)!}{(u-1)!2^{u-1}} + 2^m$$

5.4 The case of $m \equiv 2 \bmod 4$

Gold codes exist whenever m is odd or $m \equiv 2 \bmod 4$ [18]. In the case $m \equiv 2 \bmod 4$ there are still 2^m second order Boolean functions constructed using

$$h_\gamma(x_1, x_2, ..., x_m) = \mathrm{Tr}_m(\gamma x^5) \quad (5.6)$$

but the peak inner product (peak aperiodic correlation with $\tau = 0$) is $2^{(m+2)/2}$, so they can no longer be referred to as almost bent functions. Although the correlation is double that for a Kerdock code (for which it is $2^{m/2}$), the normalized value does not exceed 0.25 for $m \geq 6$. Given the potential advantage from the security viewpoint of not having a unique partition into Hadamard matrices, it seems preferable to use the Gold code for $m \equiv 2 \bmod 4$ as well. The analysis is much the same as in section 5.3.1 with two exceptions. Firstly, when $m \equiv 2 \bmod 4$, it can be observed from the proof of theorem 3 that there is no longer a bijection between the set of nonzero values of Λ and the set of nonzero values of γ. As $2^m - 1 \equiv 0 \bmod 3$ the field has cube roots of unity 1, $\omega = \alpha^{(2^m-1)/3}$, $\omega^2 = \alpha^{2(2^m-1)/3}$, and $\gamma^3\Lambda^{15} = 1$ has solutions $\gamma = \Lambda^{-5}$, $\gamma = \omega\Lambda^{-5}$ and $\gamma = \omega^2\Lambda^{-5}$. However, a vector Δ still exists for each nonzero value of γ, which allows switching to take place. Secondly, it can be observed that whereas $\delta' = 1$ for m odd in the proof of theorem 3, in this case $\delta' = 0$ for $\gamma = \Lambda^{-5}$ as $\mathrm{Tr}_m(1) = 0$ for m even. When $\delta' = 0$ for a particular value of γ it is necessary in section 5.3.2 to replace $L = K_1 \cup (K_1 + \hat{h}_{\gamma_3})$ and $L' = K_2 \cup (K_2 + \hat{h}_{\gamma_3})$ by $L = K_1 \cup (K_2 +$

$\hat{h}_{\gamma_3}$) and $L' = K_2 \cup (K_1 + \hat{h}_{\gamma_3})$. With this change, agreements on P will still correspond to disagreements on the corresponding value of $\bar{P}$ (and vice versa) and the argument works as before. The argument used to generate a partition into Hadamard matrices with each Hadamard matrix consisting of exactly one row of each Hadamard matrix of the original partition no longer works when $\delta' = 0$ for some values of γ. However, the subcode given by vectors $\hat{h}_\gamma$ is an (extended) 5th decimation of an m-sequence and so is an (extended) m-sequence [18]. Thus it and its cosets in the (extended) Gold code correspond to a partition into Hadamard matrices. Incomplete clique searches have failed to reveal any other partitions of this type, so in this case a lower bound is:

$$1 + \sum_{u=1}^{2^{m-1}} \binom{2^m}{2u} \frac{(2u-1)!}{(u-1)!2^{u-1}} + 1. \tag{5.7}$$

5.5 The case of $m \equiv 0 \bmod 4$

Ideally a code with similar properties to a Gold code (which does not exist in this case) should be used here in preference to the Kerdock code. Candidates include the Gold-like codes described in [18]. These are again constructed from a maximal length sequence and a qth decimation of a maximal length sequence and it is required that $\gcd(q, 2^m - 1) = 3$. Note that in this case $2^m - 1 \equiv 0 \mod 3$ and $\gcd(1 + 2^{\lfloor (m+2)/2 \rfloor}, 2^m - 1) = 3$. Then the second order Boolean functions are obtained as:

$$h_\gamma(x_1, x_2, ..., x_m) = \mathrm{Tr}_m(\gamma x^{1+2^{\lfloor (m+2)/2 \rfloor}}) \tag{5.8}$$

Theorem 6 *[18]. The above construction gives 2^m second order Boolean functions leading to 2^m Hadamard matrices, with the inner product of rows of distinct matrices having values* 0, $-2^{\lfloor (m+2)/2 \rfloor}$ $2^{\lfloor (m+2)/2 \rfloor}$, $2^{m/2}$, $-2^{m/2}$.

Following section 5.3.1 and attempting to solve the equation $\gamma\Lambda^{(1+2^r)} = \gamma\Lambda^{(1+2^{\lfloor (m+2)/2 \rfloor})} = 1$ it can be observed that a (unique) solution for Λ exists if and only if γ is of the form α^{3j} for a primitive element α of the field.

Again it can be seen that $\delta' = 0$, and it is necessary in section 5.3.2 to replace $L = K_1 \cup (K_1 + \hat{h}_{\gamma_3})$ and $L' = K_2 \cup (K_2 + \hat{h}_{\gamma_3})$ by $L = K_1 \cup (K_2 + \hat{h}_{\gamma_3})$ and $L' = K_2 \cup (K_1 + \hat{h}_{\gamma_3})$. The same graph can be constructed as in the m odd case, but only one third of the edges (those corresponding to functions $h_{\alpha^{3i}}(x_1, x_2, ..., x_m)$) allow switching and can be included in the matching. This make it hard to envisage a single formula for the number of matchings.

In the case $m = 4$ (which is probably the case of most interest here), the number of Hadamard matrices that can be obtained by switching in this way is easily counted as $(16 \times 5 \times 2)/2 = 80$ in addition to the original 16 Hadamard

matrices. However, for each switching of 8 rows of one matrix with 8 rows of another, there are 4 switchings consisting of 12 rows of one and 4 rows of the other, giving 5 times the number of switchings. Then the number of possible matrices is 416.

The maximum modulus of the normalized inner product is 0.5 in the case $m = 4$, which is unsatisfactory in terms of the correlation criterion. However, for each possible Hadamard matrix row only 20 of the 255 other Hadamard matrix rows achieve this value, so it is difficult to see how it could be exploited.

Chapter 6

Sequential generations

Random methods of evolving internal padding were presented in chapter 3. A better method would work along the lines of chapters 4 and 5. A set of permutations of internal padding lengths should be found so that correlations between generations do not exceed a given threshold.

The components of the internal padding vector are defined by the number L_i of zeros inserted between consecutive components of Golay pairs in an LS code. The vector is denoted $(L_1, L_2, \ldots, L_{p-1})$. It is not feasible to consider all possible permutations of the internal paddings lengths. If all the paddings are of different lengths, then the number of distinct vectors is $(p-1)!$. If two or more of the L_i are equal in length, there are fewer different arrangements of the paddings: Let one ordering of the internal padding lengths be

$$(L_1, L_2, ..., L_{p-1}) = (\underbrace{m_1, m_1, m_1}_{r_1}, \underbrace{m_2, m_2}_{r_2}, ..., \underbrace{m_q, ..., m_q}_{r_q}).$$

where

$$r_1 + r_2 + ... + r_q = p - 1$$

Then the number of distinct codes obtained by varying the internal padding is

$$\frac{(p-1)!}{\prod_{j=1}^{q} r_j!}.$$

In our typical case the padding length vector is

$$P = (0,0,0,0,1,1,1,1,2,2,2,2,3,3,3,3,4,4,4,4,5,5,5,5,6,6,6,6,7,7,7)$$

and the total number of permutations is $298,807,728,716,735,988,750,000$. A sequential (greedy) method for generating a set of compatible internal padding vectors (leading to satisfactory correlation) can be devised, and is presented here as algorithm 1.

Algorithm 1

Initialise a Hadamard matrix
Initialise a Golay pair and mate
Initialise a choice of π *vector*
Create an empty IP list of length α
Initialise a seed permutation of the internal padding vector P
Append P *to IP list*
Initialise the number of iterations i
For iterations from 1 to i *do*
    Generate a new permutation of P *randomly;*
    If P *has normalized aperiodic correlation* $\leq \sigma$ *with all internal padding vectors in the IP list then*
        Append P *to IP list;*
    End if
End for

A fixed Golay pair, Hadamard matrix and π vector are selected. A permutation of P is then selected as a seed permutation to create an LS code. A new permutation of P is generated randomly and a new LS code is created with P replaced by this new permutation of P. The peak normalized aperiodic cross-correlation between all single codewords from the first code and the corresponding codewords (using the same Hadamard matrix row) from the second are found and the maximum is recorded. If the maximum is less than or equal to a threshold σ then the new permutation of P is accepted for the second generation. If the maximum is greater than σ a new internal padding permutation is created and the previous one discarded. This process of accepting and rejecting is repeated for a set number of iterations and the correlation of the code obtained is checked with every previous code obtained from an accepted permutation of P. The total number of codes that work together (generations) can then be calculated. Results can be seen in table 6.1 for a case when $\pi = (0, 1, 1, 0, 0, 1, 1, 0, .., 1, 1, 0)$. The results are quite satisfactory. With a correlation threshold of $\sigma = 0.25$, more than 900 distinct permutations are possible. Even when the number of vectors is the same in two distinct cases, the actual vectors are not identical.

In table 6.2 the computations are repeated using a vector $\pi = (1, 1, ..., 1, 1)$. This ensures that the number of coincidences of Golay pairs does not exceed a specified value v. This suggests another method of generating sets of compatible internal padding vectors by monitoring the number of exact coincidences. This will be dealt with in detail in chapter 7.

iterations	σ	*run 1*	*run 2*	*run 3*
100,000	0.1	4	4	5
100,000	0.2	95	97	100
100,000	0.25	844	759	844
100,000	0.5	9980	7168	8549
1,000,000	0.1	6	5	5
1,000,000	0.2	128	112	128
1,000,000	0.25	928	712	867
100,000,000	0.1	7	6	5

Table 6.1: *The number of evolutions generated using a greedy algorithm with a random internal padding generator, using three different seed permutations of P in run 1, run 2 and run 3 respectively.*

σ	v	number of evolutions
0.03125	1	1
0.0625	2	1
0.09375	3	2
0.125	4	5
0.15625	5	14
0.1875	6	49
0.21875	7	136
0.25	8	378
0.3125	10	1133
0.375	12	3004

Table 6.2: *The number of evolutions generated using a greedy algorithm with a random internal padding generator (10,000,000 iterations), with* $\pi = (1, 1, ..., 1, 1)$.

Chapter 7

Development of a bounding theorem

Recall the notation from section 1.7 of chapter 1. In this work it is required to construct a very long sequence P of generations of the code such that for any user u and distinct generations g_i and g_j:

$$|\theta^{(a)}_{x_u(g_i)x_u(g_j)}| \leq \chi$$

where χ is a chosen bound for the normalized aperiodic correlation.

Suppose the Golay pairs and the permutation of the internal padding lengths are fixed, but we allow g_i and g_j to be constructed with different Hadamard matrices (and perhaps different columns of a Latin square). It may then be possible to show that for some χ_1 (independent of the actual choices of the permutation of internal padding lengths and Golay pairs):

$$|\theta^{(a)}_{x_u(g_i)x_u(g_j)}| \leq \chi_1.$$

Similarly, suppose the Hadamard matrices and the permutation of the internal padding lengths are fixed, but we allow g_i and g_j to be constructed with different Golay pairs. It may be possible to show that for some χ_2 (independent of the actual choices of the permutation of internal padding lengths and Hadamard matrix rows):

$$|\theta^{(a)}_{x_u(g_i)x_u(g_j)}| \leq \chi_2.$$

Finally, suppose the Hadamard matrices and the Golay pairs are fixed, but we allow g_i and g_j to be constructed with different permutations of the internal padding lengths. It may be possible to show that for some χ_3 (independent of the actual choices of the Hadamard matrix rows and Golay pairs):

$$|\theta^{(a)}_{x_u(g_i)x_u(g_j)}| \leq \chi_3.$$

The aim of the bounding theorem is to find $\chi = F(\chi_1, \chi_2, \chi_3)$ such that if all three components of the construction are varied (different Hadamard matrices, different Golay pairs and different permutation of the internal padding lengths, then

$$|\theta^{(a)}_{x_u(g_i)x_u(g_j)}| \leq F(\chi_1, \chi_2, \chi_3).$$

It would be ideal to find a function F similar to $F(\chi_1, \chi_2, \chi_3) = \chi_1\chi_2\chi_3$ but such an aspiration is far too optimistic. In this chapter a bounding theorem will be proved with $F(\chi_1, \chi_2, \chi_3) = \max\{\chi_1, \chi_2, \chi_3\}$. Even to achieve this result, considerable care must be taken with the way that χ_1, χ_2 and χ_3 are computed.

7.1 Definition of χ_1

Let $p = 2^m$ and consider the $2^{2\lceil \frac{m}{2} \rceil - 1}$ different $p \times p$ Hadamard matrices constructed using bent, almost bent or other second order Boolean functions as in sections 5.1, 5.2, 5.4 and 5.5. Let $\mathbf{r}^{(i)}_j$ represent row j of the ith matrix:

$$H_i = \begin{bmatrix} \mathbf{r}^{(i)}_1 \\ \mathbf{r}^{(i)}_2 \\ \vdots \\ \mathbf{r}^{(i)}_p \end{bmatrix}, \ i = 1, 2, ..., 2^{2\lceil \frac{m}{2} \rceil - 1}.$$

Let $B = 2^{\lfloor \frac{m}{2} \rfloor}/2^m$ (when a Kerdock code is used to define the Hadamard matrices), $B = 2^{\lfloor \frac{m+1}{2} \rfloor}/2^m$ (when a Gold code with m odd is used to define the Hadamard matrices) or $B = 2^{\lfloor \frac{m+2}{2} \rfloor}/2^m$ (when a Gold code with $m \equiv 2 \bmod 4$ or a Gold-like code with $m \equiv 0 \bmod 4$ is used to define the Hadamard matrices).

The results of chapter 5 then imply that for
$i, j \in \{1, 2, \ldots, 2^{2\lceil \frac{m}{2} \rceil - 1}\}$ and $s, t \in \{1, 2, \ldots, p\}$:

$$\theta^{(a)}_{\mathbf{r}^{(i)}_s \mathbf{r}^{(i)}_t}(0) = 0, \quad s \neq t, \quad \forall\, i,$$

$$\theta^{(a)}_{\mathbf{r}^{(i)}_s \mathbf{r}^{(j)}_t}(0) \leq B, \quad i \neq j, \quad \forall\, s, t.$$

where $\theta^{(a)}_{\mathbf{r}^{(i)}_s \mathbf{r}^{(i)}_t}(0) = C^{(a)}_{\mathbf{r}^{(i)}_s \mathbf{r}^{(i)}_t}(0)/C^{(a)}_{\mathbf{r}^{(i)}_s \mathbf{r}^{(i)}_s}(0)$ is the normalized aperiodic cross-correlation between rows when $\tau = 0$ (i.e. the inner product of rows, scaled between 0 and 1). Use each Hadamard matrix H_i with fixed padding, a fixed Golay pair and fixed vectors π and π^* to construct an LS code. We obtain a matrix with $2p$ rows:

$$C_i = \begin{bmatrix} \mathbf{x}^{(i)}_1 \\ \mathbf{x}^{(i)}_2 \\ \vdots \\ \mathbf{x}^{(i)}_{2p} \end{bmatrix}$$

where row $\mathbf{x}_j^{(i)}$ is the jth codeword in the LS code. For this code

$$\theta^{(a)}_{\mathbf{x}_s^{(i)}\mathbf{x}_t^{(i)}}(\tau) \leq \chi_0, \quad s \neq t, \quad -\tau_{\max} < \tau < \tau_{\max},$$

$$\theta^{(a)}_{\mathbf{x}_s^{(i)}\mathbf{x}_t^{(j)}}(\tau) \leq \max\{\chi_0, B\}, \quad i \neq j, \quad -\tau_{\max} < \tau < \tau_{\max},$$

where χ_0 is a bound for the maximum normalized aperiodic correlation of the code within the correlation window. Specifically, suppose that the actual set of internal padding lengths $\{L_1, L_2, L_3, ..., L_{p-1}\}$ is given, but not their arrangement. Also, let m_p denote the maximum number of times any internal padding length is repeated. Then if we take $\chi_0 = m_p/p$, the upper bound for the correlation given by χ_0 is independent of the choice of Golay pair (and its mate), independent of π and is also independent of the actual arrangement of internal padding lengths used. The value of χ_0 can be improved if transitions and non-transitions in π are taken into account, but this seems unnecessary here as χ_0 is usually much less than B.

Let $\chi_1 = \max\{\chi_0, B\}$, then we have seen that $\theta^{(a)}_{\mathbf{x}_s^{(i)}\mathbf{x}_s^{(j)}}(\tau) \leq \chi_1$ for all s, i, j, $i \neq j$ and $-\tau_{\max} < \tau < \tau_{\max}$. The value of χ_1 observed gives a satisfactory correlation criterion between generations when $m \geq 5$.

7.2 Definition of χ_2 as $\chi_2^{(1)}$, $\chi_2^{(2)}$, or $\chi_2^{(3)}$.

Consider the compatible sets of pairs (C_0, S_0), (C_1, S_1) (i.e Golay pair, Golay pair mate) found in chapter 4. Suppose that there are η pairs in a set, and denote them:

$$G_j, \quad j \in \{1, 2, \ldots, \eta\}.$$

Let H_i be a Hadamard matrix constructed using bent or almost bent functions as in chapter 5. Let $\mathbf{x}_u(G_j, H_i)$ be a row of a LS code constructed using G_j and H_i for fixed internal padding lengths and vector π. Then the properties of LS codes combined with the results of chapter 4 show that for some fixed Hadamard matrix H_i:

$$\theta^{(a)}_{\mathbf{x}_u(G_j,H_i)\mathbf{x}_u(G_k,H_i)}(\tau) \leq \chi_2^{(1)}, \quad j \neq k, \quad -\tau_{\max} < \tau < \tau_{\max}$$

with $\chi_2^{(1)} = \sigma$ where σ is the correlation threshold. For example, $N = 16$, $\mu = 1024$ and $\sigma = 0.274$ a clique of size 22 was obtained in table 4.2. The result can be generalised to show that the clique of size 22 gives a bound $\chi_2^{(1)} = 0.277$ for *any* of the Hadamard matrices constructed using almost bent functions with $\pi = (0,1,1,0,0,1,1,\ldots,1,1,0)$ and internal padding vector $(0,0,0,0,1,1,1,1,2,2,2,2,3,3,3,3,4,4,4,4,5,5,5,5,6,6,6,6,7,7,7)$. Thus for one permutation of internal padding, $\chi_2^{(1)}$ has been made independent of the

choice of Hadamard matrix. It is considerably more challenging to show that $\chi_2^{(1)}$ can be made independent of the choice of permutation of the internal padding lengths, and this computation has not been attempted.

A slightly different bound $\chi_2^{(2)}$ might be computed, showing

$$\theta^{(a)}_{\mathbf{x}_u(G_j,H_i)\mathbf{x}_u(G_k,H_l)}(\tau) \leq \chi_2^{(2)}, \quad j \neq k, \quad i \neq l, \quad -\tau_{\max} < \tau < \tau_{\max}.$$

A computation with different matrices from the same set of Hadamard matrices, same π vector, same internal padding vector and the clique of size 22 gives $\chi_2^{(2)} = 0.3125$.

Suppose that the type of clique computation in chapter 4 is repeated for the Golay pairs (and mates) alone. Specifically, a vertex v_i is in the graph for each Golay pair G_i and its mate G'_i. An edge $(v_j v_k)$ is present if the following two conditions are satisfied. Firstly, the maximum normalized aperiodic correlation between the two Golay pairs G_j, G_k, the two Golay pair mates G'_j, G'_k and both pair/mate combinations G_j, G'_k and G'_j, G_k are at most some chosen value $\chi_2^{(3)}$, i.e.

$$\begin{aligned}
\theta^{(a)}_{G_j,G_k}(\tau) &\leq \chi_2^{(3)}, \quad j \neq k, \quad -N < \tau < N, \\
\theta^{(a)}_{G_j,G'_k}(\tau) &\leq \chi_2^{(3)}, \quad j \neq k, \quad -N < \tau < N, \\
\theta^{(a)}_{G'_j,G_k}(\tau) &\leq \chi_2^{(3)}, \quad j \neq k, \quad -N < \tau < N, \\
\theta^{(a)}_{G'_j,G'_k}(\tau) &\leq \chi_2^{(3)}, \quad j \neq k, \quad -N < \tau < N.
\end{aligned}$$

Secondly, if a Golay pair overlaps partially with two other Golay pairs (as seen in figure 7.1) the maximum sum of modulus of normalized aperiodic correlations for all combinations of the two Golay pairs G_j, G_k and the two Golay pair mates G'_j, G'_k are at most $\chi_2^{(3)}$, i.e.

$$\begin{aligned}
\text{for } j \neq k, \quad 1 \leq \tau_1 \leq N-1, -(N-1) \leq \tau_2 \leq -1, \quad \tau_1 - \tau_2 \geq N, \\
|\theta^{(a)}_{G_j,G_k}(\tau_1)| + |\theta^{(a)}_{G_j,G_k}(\tau_2)| \leq \chi_2^{(3)}, \\
|\theta^{(a)}_{G_j,G_k}(\tau_1)| + |\theta^{(a)}_{G_j,G'_k}(\tau_2)| \leq \chi_2^{(3)}, \\
|\theta^{(a)}_{G_j,G'_k}(\tau_1)| + |\theta^{(a)}_{G_j,G_k}(\tau_2)| \leq \chi_2^{(3)}, \\
|\theta^{(a)}_{G_j,G'_k}(\tau_1)| + |\theta^{(a)}_{G_j,G'_k}(\tau_2)| \leq \chi_2^{(3)}, \\
|\theta^{(a)}_{G'_j,G_k}(\tau_1)| + |\theta^{(a)}_{G'_j,G_k}(\tau_2)| \leq \chi_2^{(3)}, \\
|\theta^{(a)}_{G'_j,G_k}(\tau_1)| + |\theta^{(a)}_{G'_j,G'_k}(\tau_2)| \leq \chi_2^{(3)},
\end{aligned}$$

$$|\theta^{(a)}_{G'_j,G'_k}(\tau_1)| + |\theta^{(a)}_{G'_j,G_k}(\tau_2)| \leq \chi_2^{(3)},$$
$$|\theta^{(a)}_{G'_j,G'_k}(\tau_1)| + |\theta^{(a)}_{G'_j,G'_k}(\tau_2)| \leq \chi_2^{(3)}.$$

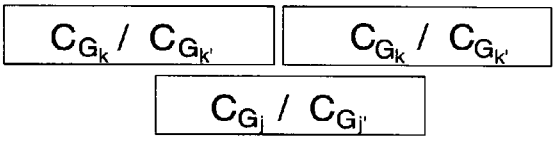

Figure 7.1: *A diagram showing how the clique search calculates any double overlaps of Golay pairs. A Golay pair component in the second codeword may overlap partially with two Golay pair components in the first codeword, possibly separated by some internal padding.*

The maximum clique in this modified graph is then computed. The results are shown in table 7.1. For $N = 16$, only a clique of size 1 is obtained for $\chi_2^{(3)} = \sigma = 0.25$. A clique of size 2 is obtained for $\chi_2^{(3)} = 0.3125$ and a clique of size 5 is obtained for $\chi_2^{(3)} = 0.375$. Note that if one of these cliques is used, the correlation bound applies to the two generations of LS code, whatever the Hadamard matrix, internal padding vector or π vector (even if the Hadamard matrix row and internal padding vector are the same in the two generations).

N	σ	*Size of clique*
16	1	384
16	0.375	5
16	0.3125	2
16	0.25	1

Table 7.1: *The clique size for various thresholds σ of peak normalized aperiodic cross-correlation using the double overlap clique search for Golay pairs. N is the length of the Golay pair.*

This third definition appears to lead to the simplest proof of a bounding theorem so, although either the number of different Golay pairs is limited or $\chi_2^{(3)}$ is somewhat larger than is desirable, $\chi_2 = \chi_2^{(3)}$ will be taken.

If any change of Golay pair is to guarantee correlations of at most σ, the new clique search method shown in this chapter must be used. The number of Golay pairs is drastically reduced from the results in chapter 4. To *guarantee* normalized correlation of no more than 0.25 using changes of Golay pairs alone

the clique size reduces from 8 to 1, giving no evolution of the Golay pairs at this threshold. More practically, if the internal padding vector is forced to change to guarantee satisfactory correlation, the results suggest a sensible choice is to use the clique of 5 Golay pairs. In chapter 4 a similarly sensible decision to use the clique of 22 Golay pairs was proposed. The clique of size 5 found here will be evaluated in chapter 10.

7.3 Definition of χ_3 as $\chi_3^{(1)}$ or $\chi_3^{(2)}$

Consider the permutations of internal padding lengths generated sequentially in chapter 6. Suppose that there are ϱ of them, and denote them:

$$P_k, \quad k \in \{1, 2, \ldots, \varrho\}.$$

Let H_i be a Hadamard matrix constructed using bent or almost bent functions as in chapter 5. Let $\mathbf{x}_u(P_k, H_i)$ be a row of a LS code constructed using P_k and H_i for a fixed Golay pair and fixed vector π. Then the properties of LS codes combined with the results of chapters 5 and 6 show that:

$$\theta^{(a)}_{\mathbf{x}_u(P_k,H_i)\mathbf{x}_u(P_z,H_i)}(\tau) \leq \chi_3^{(1)}, \quad k \neq z, \quad -\tau_{\max} < \tau < \tau_{\max}.$$

where $\chi_3^{(1)}$ is the correlation threshold used in the sequential generation. It remains to be shown that $\chi_3^{(1)}$ is independent of π and of the Golay pair and Hadamard matrix used.

An alternative approach is to let IP_c be the maximum number of permutations of internal paddings possible such that for any pair of permutations there are at most c *coincidences*. A coincidence occurs when a Golay pair in the code with the first internal padding permutation is precisely aligned with a Golay pair in the code with the second internal padding permutation (i.e. with $\tau = 0$). Note that it is not necessary that these two pairs correspond to the same column of the Hadamard matrix. Let the Hadamard matrix be a $p \times p$ matrix and define $\chi_3^{(2)} = c/p$. Then for any fixed Golay pair and Hadamard matrix:

$$\theta^{(a)}_{\mathbf{x}_u(P_k,H_i)\mathbf{x}_u(P_z,H_i)}(\tau) \leq \chi_3^{(2)}, \quad k \neq z, \quad -\tau_{\max} < \tau < \tau_{\max}$$

as non-coincident Golay pairs contribute nothing to the aperiodic correlation. Clearly $\chi_3^{(2)}$ is independent of the Hadamard matrix and Golay pair chosen. Computations similar to those in Chapter 6 show that if $p = 32$ then $\mathrm{IP}_8 \geq 378$, $\mathrm{IP}_{10} \geq 1133$ and $\mathrm{IP}_{12} \geq 3004$.

This second definition appears to lead to the simplest proof of a bounding theorem so $\chi_3 = \chi_3^{(2)}$ will be taken initially.

7.4 Hybrid methods

7.4.1 Use of Latin squares to increase the number of generations

In this section the notation of section 7.1 will be used. Now for each code C_i consider a $2p \times 2p$ Latin square; let $P_1^{(i)}$ be the identity permutation on column 1 of the Latin square and for $j = 2, 3 \ldots, 2p$, let $P_j^{(i)}$ be the permutation of column 1 of the Latin square that gives column j. Then the $2p \times 2^{2\lceil \frac{m}{2} \rceil - 1}$ generations of code

$$P_1^{(1)}(C_1), P_2^{(1)}(C_1), P_3^{(1)}(C_1), \ldots, P_{2p}^{(1)}(C_1), P_1^{(2)}(C_2), \ldots, P_{2p}^{(2)}(C_2), \ldots$$

$$\ldots, P_1^{(2^{2\lceil \frac{m}{2} \rceil - 1})}(C_{2^{2\lceil \frac{m}{2} \rceil - 1}}), \ldots, P_{2p}^{(2^{2\lceil \frac{m}{2} \rceil - 1})}(C_{2^{2\lceil \frac{m}{2} \rceil - 1}})$$

have the property that for $i, j \in \{1, 2, \ldots 2p \times 2^{2\lceil \frac{m}{2} \rceil - 1}\}$ and all $s \in 1, 2, \ldots 2p$,

$$\theta^{(a)}_{\mathbf{x}_s^{(i)} \mathbf{x}_s^{(j)}}(\tau) \leq \max\{\chi_0, B\} = \chi_1, \quad i \neq j, \quad -\tau_{\max} < \tau < \tau_{\max}.$$

As for relevant codes $\chi_0 \leq B$ and B itself can give a satisfactory correlation criterion, this hybrid of the Latin square construction with a suitable Hadamard matrix construction from chapter 5 gives $2p \times 2^{2\lceil \frac{m}{2} \rceil - 1}$ generations with satisfactory correlation.

7.4.2 A bounding theorem

Theorem 7 *Suppose that a set of generations of LS codes is created with a hybrid method by using all combinations of*

1. *sets of codes obtained using all Hadamard matrices constructed by bent or almost bent functions and Latin squares, with maximum normalized aperiodic correlation* χ_1,
2. *the clique of Golay pairs generated using* $\chi_2 = \chi_2^{(3)}$,
3. *the set of* IP_c *internal padding permutations giving* $\chi_3 = c/p$.

Then the normalized aperiodic correlation for any two codewords $\mathbf{x}_s^{(i)}$, $\mathbf{x}_s^{(j)}$ *from distinct generations* g_i *and* g_j *satisfies*

$$\theta^{(a)}_{\mathbf{x}_s^{(i)} \mathbf{x}_s^{(j)}}(\tau) \leq \max\{\chi_1, \chi_2, \chi_3\}, \quad i \neq j, \quad -\tau_{\max} < \tau < \tau_{\max}.$$

Proof If two codes are created using distinct Golay pairs then

$$\theta^{(a)}_{\mathbf{x}_s^{(i)}\mathbf{x}_s^{(j)}}(\tau) \leq \chi_2, \quad i \neq j, \quad -\tau_{\max} < \tau < \tau_{\max}.$$

Then attention can be restricted to codes generated by the same Golay pair. The normalized aperiodic correlation for any two codewords $\mathbf{x}_s^{(i)}$, $\mathbf{x}_s^{(j)}$ from distinct internal padding permutations from the set of IP_c permutations satisfies

$$\theta^{(a)}_{\mathbf{x}_s^{(i)}\mathbf{x}_s^{(j)}}(\tau) \leq \chi_3, \quad i \neq j, \quad -\tau_{\max} < \tau < \tau_{\max}.$$

Restricting attention to codes generated by the same Golay pair and the same permutation of internal padding lengths gives:

$$\theta^{(a)}_{\mathbf{x}_s^{(i)}\mathbf{x}_s^{(j)}}(\tau) \leq \chi_1, \quad i \neq j, \quad -\tau_{\max} < \tau < \tau_{\max},$$

as shown in section 7.4.1.

Chapter 8

Evolution

In this chapter methods that enable the LS code to evolve are presented. These methods repeatedly change the code from one of the options identified in previous chapters to another option.

8.1 Pseudo-code and algorithms

This section contains pseudo-code of the key mechanisms used for evolution. In the first two mechanisms a measure of aperiodic correlation for two different internal padding vectors is referred to. This measure is the number of coincidences of Golay pair components. Coincidences occur when a Golay pair component of a codeword shifts exactly on to the next component of the other codeword. If there are at most c coincidences then the aperiodic correlation of the codewords is at most $\chi_3 = 2cN$.

Mechanism 1

Initialise x Hadamard matrices (Had[x]), *all pairs with scaled inner product* $\leq \chi_1$
Initialise y Golay pairs and their mates (Golay[y]), *all pairs with normalized aperiodic correlation* $\leq \chi_2$
Initialise z choices of internal padding vectors (IP[z]) , *all pairs giving normalized aperiodic correlation* $\leq \chi_3$
Initialise the number of generations i
For generations from 1 to i do
    Generate a random number $n \in \{1, 2, \ldots, x\}$
    Had = Had[n];
    Generate a random number $n \in \{1, 2, \ldots, y\}$
    Golay = Golay[n];
    Generate a random number $n \in \{1, 2, \ldots, z\}$
    IP = IP[n];
    Create an LS code from the three components;

End for

In mechanism 1 all components of the construction of LS codes are randomly selected (with repetition allowed) after each generation.

Many variations of mechanism 1 are possible. Components can be selected sequentially, randomly without repetition, randomly with repetition or even with some (but not all) components fixed. Mechanism 1 seems to be the least predictable and will be the basis of most of the study in chapter 10. However if sets of Hadamard matrices, Golay pairs and internal padding vectors are chosen so that correlation between pairs of generations is small (as in earlier chapters) then the values of x, y, and z may be fairly small. A third party observer might build up knowledge of all sets of components in use and capitalize on this knowledge to more effectively predict the codeword in use. Mechanism 2 (referred to here as a *recency mechanism*) is designed to prevent this by permitting a much larger value of z.

Mechanism 2

Initialise x Hadamard matrices (Had[x]), *all pairs with scaled inner product* $\leq \chi_1$
Initialise y Golay pairs and their mates (Golay[y]), *all pairs with normalized aperiodic correlation* $\leq \chi_2$
/ Comment: any feasible internal padding vector is allowed, i.e as defined in chapter 2 */*
Create an empty recency list of length α
Initialise a default number of attempts β
For generations from 1 to i do
    Generate a random number $n \in \{1, 2, \ldots, x\}$
    Had = Had[n];
    Generate a random number $n \in \{1, 2, \ldots, y\}$
    Golay = Golay[n];
    corr_OK = false;
    iteration_number = 1;
    While (iteration_number $\leq \beta$ *and corr_OK = false) do*
        Generate a feasible internal padding vector IP *randomly*
        If IP *has normalized aperiodic correlation* $\leq \chi_3$ *with all internal padding vectors in the recency list then*
            corr_OK = true;
            If recency list full then
                delete oldest internal padding vector;
            End if
            Append IP *to recency list*

```
        Else
            If iteration no. = 1 then
                best IP so far = IP;
            Else if no. coincidences(IP)≤ no. coincidences(best IP so far) then
                best IP so far = IP;
            End if
            If iteration_number = β and corr_OK = false then
                If recency list full then
                    delete oldest internal padding vector;
                End if
                Append best IP so far to recency list;
                IP = best IP so far;
            End if
        End if
        iteration_number = iteration_number + 1;
    End while
    Create an LS code from the three components: Had, Golay and IP;
End for
```

In mechanism 2, the Hadamard matrix row and the Golay pair are randomly selected (with repetition) after each generation. The internal padding vector is constrained only by its correlation with the previous α internal padding vectors. Then (provided that α is not too large) the majority of feasible internal padding vectors may be selected. Thus, provided the choice can be made, the correlation between close generations is guaranteed by the choice of internal padding vector. The correlation between generations that are not close may sometimes be larger, but the probability of this happening is small and there is no way to predict which generations have excessive correlation with the current generation.

Table 8.1 shows the percentage of accepted randomly generated internal padding vectors for single trials ($\beta = 1$) with various recency list sizes. It is important to note that using different sizes of recency list creates different latency times on the system. If a recency list of size 50 were used in a given system, then the latency time in creating a new internal padding vector is considerably longer than when using a list of length 1. With a recency list of length 1, almost all new internal padding vectors are accepted (96.4%), whereas with a recency list of length 50, only 26.8% are accepted.

Table 8.2 shows the percentage of trials rejected when $\beta > 1$. Here a single trial is a maximum of β iterations. The mean and maximum correlation for the rejected cases is also shown.

Recency list size	*No. accepted*	*No. rejected*	*% Accepted*
1	964148	35852	96.4
10	906980	93020	90.7
15	798369	201631	79.8
20	551903	448097	55.2
30	425037	574963	42.5
40	332382	667618	33.2
50	268105	731895	26.8
60	197153	808247	19.2

Table 8.1: *The number of accepted and rejected internal padding length vectors for $p = 32$ with at most eight coincidences (out of 32) with any vector within the recency list. One million vectors are generated in each case.*

It can be seen that a recency list of size up to 50, with β between 5 and 10 seems satisfactory for $p = 32$. The number of outliers with a correlation approaching 0.5 is in fact very small.

Recency list size	β	*% Rejected after β iterations*	*Mean normalized correlation for those rejected*	*Maximum normalized correlation for those rejected*
10	10	0.090	0.294	0.34375
20	10	0.250	0.293	0.4065
30	10	0.543	0.297	0.375
40	10	0.798	0.298	0.4065
50	10	0.920	0.302	0.5
60	10	1.106	0.304	0.5
10	5	0.01	0.29256	0.34375
20	5	0.12	0.29484	0.375
30	5	0.49	0.29744	0.4375
40	5	1.11	0.2921	0.40625
50	5	1.72	0.29239	0.46875
60	5	2.106	0.302	0.5
10	2	4.204	0.293	0.375
20	2	8.536	0.298	0.40625
30	2	18.594	0.297	0.46875
40	2	28.745	0.301	0.375
50	2	45.998	0.280	0.46825
60	2	50.737	0.300	0.5

Table 8.2: *The percentage of rejected trials after β iterations for $p = 32$. A trial is rejected if each of the β vectors generated has more than eight coincidences (out of 32) with some vectors within the recency list. For the rejected cases the mean (best of β trials) and maximum normalized correlation is shown.*

Mechanism 3

Initialise a Hadamard matrix (Had)
Initialise X *choices of column permutation*
Initialise the number of random switching operations Y *for the Hadamard matrix*
Initialise the total number of rows of the Hadamard matrix (Z)
For j from 1 *to X do*
    For k from 1 *to Y do*
        *Choose first 2nd order Boolean function randomly; /*Gold or Gold-like cases */*
        Choose second 2nd order Boolean function randomly;
        Choose K_1 *or* K_2 *if choices are distinct; /* See section 5.3.2 */*
        *Construct matrix; /*as in section 5.3.2, 5.4 or 5.5 */*
        Permute columns of constructed matrix using permutation j;
        For i in 1 to Z do
            Append row i of matrix to key for user i;
        End for
    End for
End for
For I in 1 to Z do
    Distribute key for user I privately to user I;
End for
/ The theoretical correlation results of Chapter 5 only apply to matrices constructed in the inner loop (i.e for a fixed value of j) */*

Mechanism 3 presents a method for generating Hadamard matrix rows and distributing the resulting sequences of rows as keys. Apart from parameters, the inputs to the algorithm are the given Hadamard matrix, a set of choices of column permutations and the second order Boolean functions as constructed in chapter 5. The algorithm generates new Hadamard matrices using the second order Boolean functions and after Y iterations a column permutation is applied. When the Hadamard matrix is constructed, the appropriate Hadamard matrix row is appended to each user's key. At the end of the computation the keys are distributed to the users privately. This algorithm is important as the Hadamard matrix row is the only piece of information not shared amongst all users of the system.

Chapter 9

Application of transform methods

Transform methods can be applied to identify orthogonal properties in a selected transform domain. A wide variety of transforms are available for this purpose. In this thesis the set of transforms studied is limited, and has been selected by inspection. Some transforms will exhibit trends and patterns which would lead a third party to investigate certain parameters and increase knowledge of the codeword construction.

9.1 Identifying code properties in the transform domain

Suppose that a codeword is received, with no knowledge of the code type, length or detailed structure. Such lack of knowledge would make the task of determining the actual codeword harder. The following procedure could be carried out to gather this knowledge. Firstly, the sequence could be analysed in the time domain. The codeword length will be apparent from the periodic nature of the sequence. Secondly, a Fourier transform provides the spectral content of the codeword, which could be exploited by the third party to characterize the codeword. Finally, detailed properties of the code could be identified in the transform domain. For LS codes the Hadamard transform appears the most appropriate for this final purpose.

In the remainder of this chapter attention will focus on the use of a Hadamard transform (of the same order as the size of the Golay pair component) on the assumption that the code is known to be an LS code.

9.2 Hadamard transform

LS codes are constructed using Hadamard matrices, therefore a good starting point is the Hadamard transform. The Hadamard transform of an entire $\{+1, -1\}$ codeword (or of an entire $\{0, +1, -1\}$ codeword) can be found by pre-multiplication of the vector (of length n) by an $n \times n$ Hadamard matrix. The Hadamard transform obtained can then be compared with the Hadamard transforms of other vectors to find patterns and possible ways to elucidate the various components used to create the original codewords. Note that the Hadamard transform exists whenever a Hadamard matrix exists. Fast Hadamard transforms are currently only available for Hadamard matrices of size $2^a \times 2^a$ ($a \geq 0$). In this work only Hadamard matrices of Sylvester form are used for the transform, but this does not appear to cause any real restriction in their application.

9.2.1 Identifying the Golay pair and mate used

It will be shown in the following sections that is possible to identify the Golay pair and mate used to construct the LS code using a smaller $N \times N$ Hadamard matrix transform, where N is the length of each Golay pair component (C or S). In fact it will be shown by inspection that it is possible to determine all components of the construction using this transform.

Finding C and S components

A C component can be found by shifting an LS codeword through a Hadamard transform. The amplitudes in the transform domain can be used to identify the component. Thus $H\mathbf{x}$ is calculated, where the vector $\mathbf{x}$ consists of N consecutive chips of a received LS codeword. In this way it is possible to identify the Golay pair and mate used in the construction of the LS code. Consider first components of length 16 and a 16×16 Hadamard transform. Table 9.1 shows how the C component can be found.

The eighth row of table 9.1 arises from transforming eight bits of external padding concatenated with the first eight bits of the first component C. The sixteenth row in table 9.1 arises from 16 bits of a C component and it will be shown that it identifies the C component of a Golay pair. It can be see that when a component has been located the Hadamard transform vector contains only entries of ± 4.

For the Golay pair obtained using the doubling construction (described in section 2.2) and a Hadamard matrix of Sylvester form, it is possible to describe the transform vector theoretically. Here (C_1, S_1) denotes the transpose of a Golay pair of length $1 + 1 = 2$ and in general (C_i, S_i) denotes the transpose of a

1	-1	-1	1	-1	1	1	-1	-1	1	1	-1	1	-1	-1	1
2	0	-2	0	-2	0	2	0	-2	0	2	0	2	0	-2	0
3	-1	-1	-1	-3	1	1	1	-3	1	1	1	3	-1	-1	-1
2	2	2	-2	-2	-2	-2	2	-2	-2	-2	2	2	2	2	-2
3	-3	1	3	-1	1	-3	-1	-3	3	-1	-3	1	-1	3	1
4	2	-4	2	0	-2	0	-2	-4	-2	4	-2	0	2	0	2
3	-1	-1	-5	3	-1	-1	3	-3	1	1	5	-3	1	1	-3
4	**0**	**4**	**0**	**0**	**4**	**0**	**-4**	**-4**	**0**	**-4**	**0**	**0**	**-4**	**0**	**4**
5	-1	-1	5	-1	-3	5	-1	-3	-1	-1	-3	-1	5	-3	-1
6	0	-6	0	-2	4	2	4	-2	0	2	0	-2	-4	2	-4
7	-1	-1	-5	-7	1	1	-3	-1	-1	-1	3	1	1	1	5
6	2	6	-2	-2	-6	-2	6	-2	2	-2	-2	-2	2	-2	-2
5	-1	3	5	1	3	-9	1	1	-5	-1	1	-3	-1	3	-3
4	2	-4	2	4	-6	-4	-6	4	2	-4	2	-4	2	4	2
5	-3	-3	-3	9	1	1	1	1	1	1	-7	-3	-3	-3	5
4	**4**	**4**	**-4**	**4**	**4**	**4**	**-4**	**4**	**-4**	**4**	**4**	**-4**	**4**	**-4**	**-4**
3	-3	5	3	1	-1	7	1	5	3	-5	5	-1	-7	1	-1
2	4	-2	4	-2	4	2	4	6	-4	-6	-4	2	4	-2	4
1	-3	-3	-3	-5	-1	-1	-1	7	3	3	-5	-3	1	1	-7
2	2	2	-2	-2	-2	-2	2	6	-2	6	2	2	-6	2	6
1	-1	3	1	1	-1	-5	1	3	5	1	3	3	5	-7	3
0	2	0	2	4	-2	-4	-2	0	-2	0	-2	4	-6	-4	-6
1	-3	-3	1	5	1	1	-3	3	-1	-1	3	7	3	3	-1
0	**4**	**0**	**-4**	**4**	**0**	**4**	**0**	**0**	**4**	**0**	**-4**	**4**	**0**	**4**	**0**
-1	-3	5	-1	1	3	3	1	-1	-3	5	-1	1	3	3	1
-2	4	2	4	-2	0	2	0	-2	4	2	4	-2	0	2	0
-3	-3	-3	1	-1	-1	-1	3	-3	-3	-3	1	-1	-1	-1	3
-2	2	-2	-2	-2	2	-2	-2	-2	2	-2	-2	-2	2	-2	-2
-1	-3	1	-1	-1	-3	1	-1	-1	-3	1	-1	-1	-3	1	-1
0	2	0	2	0	2	0	2	0	2	0	2	0	2	0	2
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1

Table 9.1: *Using a* 16×16 *Hadamard matrix in a Hadamard transform to locate and identify the* C *component used in an LS code of length 1296.*

Golay pair of length $i + i = 2i$. Recall from chapter 2 that sequences with length a power of 2 can be obtained by using the fact that if the generating functions $A(z), B(z)$ represent a Golay pair of length N then a Golay pair $C(z), D(z)$ of length $2N$ can be created:

$$C(z) = A(z) + z^N B(z), \quad D(z) = A(z) - z^N B(z). \tag{9.1}$$

Starting from:

$$H_1C_1 = C_1 = 1 \qquad H_1S_1 = S_1 = 1$$

$$H_2C_2 = \begin{pmatrix} C_1 + S_1 \\ C_1 - S_1 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \end{pmatrix} \qquad H_2S_2 = \begin{pmatrix} C_1 - S_1 \\ C_1 + S_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \end{pmatrix}$$

$$H_4C_4 = \begin{pmatrix} 2C_1 \\ 2C_1 \\ 2S_1 \\ -2S_1 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \\ 2 \\ -2 \end{pmatrix} \qquad H_4S_4 = \begin{pmatrix} 2S_1 \\ -2S_1 \\ 2C_1 \\ 2C_1 \end{pmatrix} = \begin{pmatrix} 2 \\ -2 \\ 2 \\ 2 \end{pmatrix}$$

$$H_8C_8 = \begin{pmatrix} 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ -2C_1 + 2S_1 \\ -2C_1 - 2S_1 \end{pmatrix} = \begin{pmatrix} 4 \\ 0 \\ 4 \\ 0 \\ 0 \\ 4 \\ 0 \\ -4 \end{pmatrix} \qquad H_8S_8 = \begin{pmatrix} 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ -2C_1 + 2S_1 \\ -2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 4 \\ 0 \\ -4 \\ 4 \\ 0 \\ 4 \\ 0 \end{pmatrix}$$

$$H_{16}C_{16} = \begin{pmatrix} 4C_1 \\ 4C_1 \\ 4S_1 \\ -4S_1 \\ 4C_1 \\ 4C_1 \\ 4S_1 \\ -4S_1 \\ 4S_1 \\ -4S_1 \\ 4C_1 \\ 4C_1 \\ -4S_1 \\ 4S_1 \\ -4C_1 \\ -4C_1 \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \\ 4 \\ -4 \\ 4 \\ 4 \\ 4 \\ -4 \\ 4 \\ -4 \\ 4 \\ 4 \\ -4 \\ 4 \\ -4 \\ -4 \end{pmatrix} \qquad H_{16}S_{16} = \begin{pmatrix} 4S_1 \\ -4S_1 \\ 4C_1 \\ 4C_1 \\ -4S_1 \\ 4S_1 \\ -4C_1 \\ -4C_1 \\ 4C_1 \\ 4C_1 \\ 4S_1 \\ -4S_1 \\ 4C_1 \\ 4C_1 \\ 4S_1 \\ -4S_1 \end{pmatrix} = \begin{pmatrix} 4 \\ -4 \\ 4 \\ 4 \\ -4 \\ 4 \\ -4 \\ -4 \\ 4 \\ 4 \\ 4 \\ -4 \\ 4 \\ 4 \\ 4 \\ -4 \end{pmatrix}$$

$$H_{32}C_{32} = \begin{pmatrix} 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \end{pmatrix} = \begin{pmatrix} 8 \\ 0 \\ 8 \\ 0 \\ 0 \\ 8 \\ 0 \\ -8 \\ 8 \\ 0 \\ 8 \\ 0 \\ 0 \\ 8 \\ 0 \\ -8 \\ 0 \\ 8 \\ 0 \\ -8 \\ 8 \\ 0 \\ 8 \\ 0 \\ 0 \\ -8 \\ 0 \\ 8 \\ -8 \\ 0 \\ -8 \\ 0 \end{pmatrix} \qquad H_{32}S_{32} = \begin{pmatrix} 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 8 \\ 0 \\ -8 \\ 8 \\ 0 \\ 8 \\ 0 \\ 0 \\ -8 \\ 0 \\ 8 \\ -8 \\ 0 \\ -8 \\ 0 \\ 8 \\ 0 \\ 8 \\ 0 \\ 0 \\ 8 \\ 0 \\ -8 \\ 8 \\ 0 \\ 8 \\ 0 \\ 0 \\ 8 \\ 0 \\ -8 \end{pmatrix}$$

If we consider $n > 2$ (where n is a power of 2), it is possible, using the doubling construction of Golay pairs, to find the Hadamard transform as shown in equation 9.2.

$$H_nC_n = \begin{pmatrix} 2H_{\frac{n}{4}}C_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}C_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}S_{\frac{n}{4}} \\ -2H_{\frac{n}{4}}S_{\frac{n}{4}} \end{pmatrix} \qquad H_nS_n = \begin{pmatrix} 2H_{\frac{n}{4}}S_{\frac{n}{4}} \\ -2H_{\frac{n}{4}}S_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}C_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}C_{\frac{n}{4}} \end{pmatrix} \tag{9.2}$$

This can be shown from the following:

$$H_nC_n = \begin{pmatrix} H_{\frac{n}{2}}C_{\frac{n}{2}} + H_{\frac{n}{2}}S_{\frac{n}{2}} \\ H_{\frac{n}{2}}C_{\frac{n}{2}} - H_{\frac{n}{2}}S_{\frac{n}{2}} \end{pmatrix} \qquad H_nS_n = \begin{pmatrix} H_{\frac{n}{2}}C_{\frac{n}{2}} - H_{\frac{n}{2}}S_{\frac{n}{2}} \\ H_{\frac{n}{2}}C_{\frac{n}{2}} + H_{\frac{n}{2}}S_{\frac{n}{2}} \end{pmatrix} \tag{9.3}$$

$$H_{\frac{n}{2}}C_{\frac{n}{2}} = \begin{pmatrix} H_{\frac{n}{4}}C_{\frac{n}{4}} + H_{\frac{n}{4}}S_{\frac{n}{4}} \\ H_{\frac{n}{4}}C_{\frac{n}{4}} - H_{\frac{n}{4}}S_{\frac{n}{4}} \end{pmatrix} \qquad H_{\frac{n}{2}}S_{\frac{n}{2}} = \begin{pmatrix} H_{\frac{n}{4}}C_{\frac{n}{4}} - H_{\frac{n}{4}}S_{\frac{n}{4}} \\ H_{\frac{n}{4}}C_{\frac{n}{4}} + H_{\frac{n}{4}}S_{\frac{n}{4}} \end{pmatrix} \tag{9.4}$$

By substituting equations 9.4 into 9.3

$$H_nC_n = \begin{pmatrix} H_{\frac{n}{4}}C_{\frac{n}{4}} + H_{\frac{n}{4}}S_{\frac{n}{4}} + H_{\frac{n}{4}}C_{\frac{n}{4}} - H_{\frac{n}{4}}S_{\frac{n}{4}} \\ H_{\frac{n}{4}}C_{\frac{n}{4}} - H_{\frac{n}{4}}S_{\frac{n}{4}} + H_{\frac{n}{4}}C_{\frac{n}{4}} + H_{\frac{n}{4}}S_{\frac{n}{4}} \\ H_{\frac{n}{4}}C_{\frac{n}{4}} + H_{\frac{n}{4}}S_{\frac{n}{4}} - H_{\frac{n}{4}}C_{\frac{n}{4}} + H_{\frac{n}{4}}S_{\frac{n}{4}} \\ H_{\frac{n}{4}}C_{\frac{n}{4}} - H_{\frac{n}{4}}S_{\frac{n}{4}} - H_{\frac{n}{4}}C_{\frac{n}{4}} - H_{\frac{n}{4}}S_{\frac{n}{4}} \end{pmatrix} \tag{9.5}$$

$$H_nS_n = \begin{pmatrix} H_{\frac{n}{4}}C_{\frac{n}{4}} + H_{\frac{n}{4}}S_{\frac{n}{4}} - H_{\frac{n}{4}}C_{\frac{n}{4}} + H_{\frac{n}{4}}S_{\frac{n}{4}} \\ H_{\frac{n}{4}}C_{\frac{n}{4}} - H_{\frac{n}{4}}S_{\frac{n}{4}} - H_{\frac{n}{4}}C_{\frac{n}{4}} - H_{\frac{n}{4}}S_{\frac{n}{4}} \\ H_{\frac{n}{4}}C_{\frac{n}{4}} + H_{\frac{n}{4}}S_{\frac{n}{4}} + H_{\frac{n}{4}}C_{\frac{n}{4}} - H_{\frac{n}{4}}S_{\frac{n}{4}} \\ H_{\frac{n}{4}}C_{\frac{n}{4}} - H_{\frac{n}{4}}S_{\frac{n}{4}} + H_{\frac{n}{4}}C_{\frac{n}{4}} + H_{\frac{n}{4}}S_{\frac{n}{4}} \end{pmatrix} \tag{9.6}$$

which reduce to equations 9.2.

Thus if $N = 2^i$ the values in the transform vector when H_NC_N or H_NS_N is computed are $2^{\lfloor\frac{i+1}{2}\rfloor}$ (i even) and $\{0, 2^{\lfloor\frac{i+1}{2}\rfloor}\}$ (i odd).

If one of the six operations described in chapter 4 were used to create an alternative Golay pair, the Hadamard transform would change. It should still be possible to determine the Golay pair used in the construction of the LS code. Consider the first operation; interchanging the sequences in a pair, this results in the switching of C and S and can be seen in equation 9.7.

$$H_nC_n = \begin{pmatrix} 2H_{\frac{n}{4}}S_{\frac{n}{4}} \\ -2H_{\frac{n}{4}}S_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}C_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}C_{\frac{n}{4}} \end{pmatrix} \qquad H_nS_n = \begin{pmatrix} 2H_{\frac{n}{4}}C_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}C_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}S_{\frac{n}{4}} \\ -2H_{\frac{n}{4}}S_{\frac{n}{4}} \end{pmatrix} \tag{9.7}$$

Thus the first operation presents no difficulties.

The next two operations are: i) reversing the order of the first part of the Golay pair and ii) reversing the order of the second part of the Golay pair. Both operations are presented in the following equations. It is important to note that either or both of the operations can be carried out when a new Golay pair is created (i.e. C, S could become $C, \overleftarrow{S}$ or $\overleftarrow{C}, S$ or $\overleftarrow{C}, \overleftarrow{S}$, where $\overleftarrow{C}$ is the vector C

reversed and $\overleftarrow{S}$ is the vector S reversed). Note that the way the Golay pair is constructed has changed as shown in equation 9.8.

$$\overleftarrow{C}_n = \begin{pmatrix} \overleftarrow{S}_{\frac{n}{2}} \\ \overleftarrow{C}_{\frac{n}{2}} \end{pmatrix} \qquad \overleftarrow{S}_n = \begin{pmatrix} -\overleftarrow{S}_{\frac{n}{2}} \\ \overleftarrow{C}_{\frac{n}{2}} \end{pmatrix} \tag{9.8}$$

$$H_1\overleftarrow{C}_1 = \overleftarrow{C}_1 = 1 \qquad H_1\overleftarrow{S}_1 = \overleftarrow{S}_1 = 1$$

$$H_2\overleftarrow{C}_2 = \begin{pmatrix} \overleftarrow{S}_1 + \overleftarrow{C}_1 \\ \overleftarrow{S}_1 - \overleftarrow{C}_1 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \end{pmatrix} \qquad H_2\overleftarrow{S}_2 = \begin{pmatrix} -\overleftarrow{S}_1 + \overleftarrow{C}_1 \\ -\overleftarrow{S}_1 - \overleftarrow{C}_1 \end{pmatrix} = \begin{pmatrix} 0 \\ -2 \end{pmatrix}$$

$$H_4\overleftarrow{C}_4 = \begin{pmatrix} 2\overleftarrow{C}_1 \\ -2\overleftarrow{C}_1 \\ -2\overleftarrow{S}_1 \\ -2\overleftarrow{S}_1 \end{pmatrix} = \begin{pmatrix} 2 \\ -2 \\ -2 \\ -2 \end{pmatrix} \qquad H_4\overleftarrow{S}_4 = \begin{pmatrix} 2\overleftarrow{S}_1 \\ 2\overleftarrow{S}_1 \\ -2\overleftarrow{C}_1 \\ 2\overleftarrow{C}_1 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \\ -2 \\ 2 \end{pmatrix}$$

$$H_8\overleftarrow{C}_8 = \begin{pmatrix} 2\overleftarrow{S}_1 + 2\overleftarrow{C}_1 \\ 2\overleftarrow{S}_1 - 2\overleftarrow{C}_1 \\ -2\overleftarrow{C}_1 - 2\overleftarrow{S}_1 \\ 2\overleftarrow{C}_1 - 2\overleftarrow{S}_1 \\ 2\overleftarrow{S}_1 - 2\overleftarrow{C}_1 \\ 2\overleftarrow{S}_1 + 2\overleftarrow{C}_1 \\ -2\overleftarrow{C}_1 + 2\overleftarrow{S}_1 \\ 2\overleftarrow{C}_1 + 2\overleftarrow{S}_1 \end{pmatrix} = \begin{pmatrix} 4 \\ 0 \\ -4 \\ 0 \\ 0 \\ 4 \\ 0 \\ 4 \end{pmatrix} \qquad H_8\overleftarrow{S}_8 = \begin{pmatrix} -2\overleftarrow{S}_1 + 2\overleftarrow{C}_1 \\ -2\overleftarrow{S}_1 - 2\overleftarrow{C}_1 \\ 2\overleftarrow{C}_1 - 2\overleftarrow{S}_1 \\ -2\overleftarrow{C}_1 - 2\overleftarrow{S}_1 \\ -2\overleftarrow{S}_1 - 2\overleftarrow{C}_1 \\ -2\overleftarrow{S}_1 + 2\overleftarrow{C}_1 \\ 2\overleftarrow{C}_1 + 2\overleftarrow{S}_1 \\ -2\overleftarrow{C}_1 + 2\overleftarrow{S}_1 \end{pmatrix} = \begin{pmatrix} 0 \\ -4 \\ 0 \\ -4 \\ -4 \\ 0 \\ 4 \\ 0 \end{pmatrix}$$

$$H_{16}\overleftarrow{C}_{16} = \begin{pmatrix} 4\overleftarrow{C}_1 \\ -4\overleftarrow{C}_1 \\ -4\overleftarrow{S}_1 \\ -4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 \\ 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 \\ -4\overleftarrow{S}_1 \\ -4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 \\ -4\overleftarrow{C}_1 \\ -4\overleftarrow{S}_1 \\ -4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 \\ -4\overleftarrow{C}_1 \end{pmatrix} = \begin{pmatrix} 4 \\ -4 \\ -4 \\ -4 \\ -4 \\ 4 \\ 4 \\ 4 \\ -4 \\ -4 \\ 4 \\ -4 \\ -4 \\ -4 \\ 4 \\ -4 \end{pmatrix} \qquad H_{16}\overleftarrow{S}_{16} = \begin{pmatrix} 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 \\ 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 \\ -4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 \\ 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 \\ -4\overleftarrow{C}_1 \\ -4\overleftarrow{S}_1 \\ -4\overleftarrow{S}_1 \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \\ -4 \\ 4 \\ 4 \\ 4 \\ -4 \\ 4 \\ -4 \\ 4 \\ 4 \\ 4 \\ 4 \\ -4 \\ -4 \\ -4 \end{pmatrix}$$

$$H_{32}\overleftarrow{C}_{32} = \begin{pmatrix} 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ -4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \end{pmatrix} = \begin{pmatrix} 8 \\ 0 \\ -8 \\ 0 \\ 0 \\ 8 \\ 0 \\ 8 \\ -8 \\ 0 \\ 8 \\ 0 \\ 0 \\ -8 \\ 0 \\ 8 \\ 0 \\ 8 \\ 8 \\ 0 \\ -8 \\ 0 \\ 0 \\ 8 \\ 0 \\ 8 \\ 8 \\ 0 \\ -8 \\ 0 \end{pmatrix} \qquad H_{32}\overleftarrow{S}_{32} = \begin{pmatrix} 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ -4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{C}_1 - 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \\ 4\overleftarrow{S}_1 - 4\overleftarrow{C}_1 \\ 4\overleftarrow{C}_1 + 4\overleftarrow{S}_1 \end{pmatrix} = \begin{pmatrix} 0 \\ -8 \\ 0 \\ -8 \\ -8 \\ 0 \\ 8 \\ 0 \\ 0 \\ -8 \\ 0 \\ -8 \\ -8 \\ 0 \\ 8 \\ 0 \\ -8 \\ 0 \\ 8 \\ 0 \\ 0 \\ -8 \\ 0 \\ -8 \\ 8 \\ 0 \\ -8 \\ 0 \\ 0 \\ 8 \\ 0 \\ 8 \end{pmatrix}$$

If we consider $n > 2$ (where n is a power of 2), it is possible, using the doubling construction of the reversed Golay pairs, to find the new Hadamard transform as shown in equation 9.9.

$$H_n\overleftarrow{C}_n = \begin{pmatrix} 2H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} \\ -2H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} \\ -2H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \\ -2H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \end{pmatrix} \qquad H_n\overleftarrow{S}_n = \begin{pmatrix} 2H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \\ -2H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} \end{pmatrix} \tag{9.9}$$

This can be shown from the following:

$$H_n\overleftarrow{C}_n = \begin{pmatrix} H_{\frac{n}{2}}\overleftarrow{C}_{\frac{n}{2}} + H_{\frac{n}{2}}\overleftarrow{S}_{\frac{n}{2}} \\ -H_{\frac{n}{2}}\overleftarrow{C}_{\frac{n}{2}} + H_{\frac{n}{2}}\overleftarrow{S}_{\frac{n}{2}} \end{pmatrix} \qquad H_n\overleftarrow{S}_n = \begin{pmatrix} H_{\frac{n}{2}}\overleftarrow{C}_{\frac{n}{2}} - H_{\frac{n}{2}}\overleftarrow{S}_{\frac{n}{2}} \\ -H_{\frac{n}{2}}\overleftarrow{C}_{\frac{n}{2}} - H_{\frac{n}{2}}\overleftarrow{S}_{\frac{n}{2}} \end{pmatrix} \tag{9.10}$$

$$H_{\frac{n}{2}}\overleftarrow{C}_{\frac{n}{2}} = \begin{pmatrix} H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \\ -H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \end{pmatrix} \qquad H_{\frac{n}{2}}\overleftarrow{S}_{\frac{n}{2}} = \begin{pmatrix} H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \\ -H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \end{pmatrix} \tag{9.11}$$

By substituting equations 9.11 into 9.10

$$H_n\overleftarrow{C}_n = \begin{pmatrix} H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \\ -H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \\ -H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \\ H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \end{pmatrix} \tag{9.12}$$

$$H_n\overleftarrow{S}_n = \begin{pmatrix} H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \\ -H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \\ -H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \\ H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} - H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{C}_{\frac{n}{4}} + H_{\frac{n}{4}}\overleftarrow{S}_{\frac{n}{4}} \end{pmatrix} \tag{9.13}$$

which reduce to equations 9.9.

Another two operations involve changing the sign of the first component of the Golay pair, or changing the sign of the second component of the Golay pair or changing both components of the Golay pair. This results from setting $C_1 = -1$, $C_2 = (-1, -1)^T$ and/or $S_1 = -1$, $S_2 = (-1, +1)^T$ in the construction given by the equation 9.2.

The final operation involves changing the sign of even elements in both Golay pair components. The operation is presented in the following equations. Starting from:

$$H_1\hat{C}_1 = C_1 = 1 \qquad H_1\hat{S}_1 = S_1 = 1$$

$$\hat{C}_2 = \begin{pmatrix} 1 \\ -1 \end{pmatrix} \qquad \hat{S}_2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$H_2\hat{C}_2 = \begin{pmatrix} C_1 - S_1 \\ C_1 + S_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \end{pmatrix} \qquad H_2\hat{S}_2 = \begin{pmatrix} C_1 + S_1 \\ C_1 - S_1 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \end{pmatrix}$$

$$H_4\hat{C}_4 = \begin{pmatrix} 2C_1 \\ 2C_1 \\ -2S_1 \\ +2S_1 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \\ -2 \\ 2 \end{pmatrix} \qquad H_4\hat{S}_4 = \begin{pmatrix} -2S_1 \\ 2S_1 \\ 2C_1 \\ 2C_1 \end{pmatrix} = \begin{pmatrix} -2 \\ 2 \\ 2 \\ 2 \end{pmatrix}$$

$$H_8\hat{C}_8 = \begin{pmatrix} 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \\ -2C_1 - 2S_1 \\ -2C_1 + 2S_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 4 \\ 0 \\ 4 \\ 4 \\ 0 \\ -4 \\ 0 \end{pmatrix} \qquad H_8\hat{S}_8 = \begin{pmatrix} 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \\ -2C_1 - 2S_1 \\ -2C_1 + 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \end{pmatrix} = \begin{pmatrix} 4 \\ 0 \\ -4 \\ 0 \\ 0 \\ 4 \\ 0 \\ 4 \end{pmatrix}$$

$$H_{16}\hat{C}_{16} = \begin{pmatrix} 4C_1 \\ 4C_1 \\ -4S_1 \\ 4S_1 \\ 4C_1 \\ 4C_1 \\ -4S_1 \\ 4S_1 \\ -4S_1 \\ 4S_1 \\ 4C_1 \\ 4C_1 \\ 4S_1 \\ -4S_1 \\ -4C_1 \\ -4C_1 \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \\ -4 \\ 4 \\ 4 \\ 4 \\ -4 \\ 4 \\ -4 \\ 4 \\ 4 \\ 4 \\ 4 \\ -4 \\ -4 \\ -4 \end{pmatrix} \qquad H_{16}\hat{S}_{16} = \begin{pmatrix} -4S_1 \\ 4S_1 \\ 4C_1 \\ 4C_1 \\ 4S_1 \\ -4S_1 \\ -4C_1 \\ -4C_1 \\ 4C_1 \\ 4C_1 \\ -4S_1 \\ 4S_1 \\ 4C_1 \\ 4C_1 \\ -4S_1 \\ 4S_1 \end{pmatrix} = \begin{pmatrix} -4 \\ 4 \\ 4 \\ 4 \\ 4 \\ -4 \\ -4 \\ -4 \\ 4 \\ 4 \\ -4 \\ 4 \\ 4 \\ 4 \\ -4 \\ 4 \end{pmatrix}$$

$$H_{32}\hat{C}_{32} = \begin{pmatrix} 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 8 \\ 0 \\ 8 \\ 8 \\ 0 \\ -8 \\ 0 \\ 0 \\ 8 \\ 0 \\ 8 \\ 8 \\ 0 \\ -8 \\ 0 \\ 8 \\ 0 \\ -8 \\ 0 \\ 0 \\ 8 \\ 0 \\ 8 \\ 8 \\ 0 \\ -8 \\ 0 \\ 0 \\ 8 \\ 0 \\ 8 \\ 8 \\ 0 \\ -8 \\ 0 \end{pmatrix} \qquad H_{32}\hat{S}_{32} = \begin{pmatrix} 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ -4C_1 - 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 + 4S_1 \\ 4C_1 - 4S_1 \\ -4C_1 - 4S_1 \\ -4C_1 + 4S_1 \end{pmatrix} = \begin{pmatrix} 8 \\ 0 \\ -8 \\ 0 \\ 0 \\ 8 \\ 0 \\ 8 \\ -8 \\ 0 \\ 8 \\ 0 \\ 0 \\ -8 \\ 0 \\ -8 \\ 0 \\ 8 \\ 0 \\ 8 \\ 8 \\ 0 \\ -8 \\ 0 \\ 0 \\ 8 \\ 0 \\ 8 \\ 8 \\ 0 \\ -8 \\ 0 \end{pmatrix}$$

If we consider $n > 4$ (where n is a power of 2), it is possible, using the doubling construction of Golay pairs, to find the Hadamard transform as shown in equation 9.14.

$$H_n\hat{C}_n = \begin{pmatrix} 2H_{\frac{n}{4}}\hat{C}_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}\hat{C}_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}\hat{S}_{\frac{n}{4}} \\ -2H_{\frac{n}{4}}\hat{S}_{\frac{n}{4}} \end{pmatrix} \qquad H_n\hat{S}_n = \begin{pmatrix} 2H_{\frac{n}{4}}\hat{S}_{\frac{n}{4}} \\ -2H_{\frac{n}{4}}\hat{S}_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}\hat{C}_{\frac{n}{4}} \\ 2H_{\frac{n}{4}}\hat{C}_{\frac{n}{4}} \end{pmatrix} \tag{9.14}$$

Equation 9.14 follows the same pattern as equation 9.2. The difference comes

from altering C_2 and S_2. This is because each new Golay pair created is otherwise altered in the same way.

Thus if $N = 2^i$ then for each of the operations the values in the transform vector are again $2^{\lfloor \frac{i+1}{2} \rfloor}$ (i even) and $\{0, 2^{\lfloor \frac{i+1}{2} \rfloor}\}$ (i odd).

Every Golay pair component created using the 6 operations has a distinct transform vector. This distinct transform vectors allows the Golay pair components to be identified. Tables 9.2, 9.3, 9.4, 9.5, 9.6 and 9.7 clarify when this occurs.

C	C^T	HC
C_1	$+$	$+1$
$-C_1$	$-$	-1

Table 9.2: *A comparison of Hadamard transform vectors for Golay pair components of length* 2^0.

C	C^T	HC
C_2	$++$	$(+2, 0)$
$-C_2$	$--$	$(-2, 0)$
$\hat{C}_2$	$+-$	$(0, +2)$
$-\hat{C}_2$	$-+$	$(0, -2)$

Table 9.3: *A comparison of Hadamard transform vectors for Golay pair components of length* 2^1.

C	C^T	HC
C_4	$+++-$	$(+2,+2,+2,-2)$
$-C_4$	$---+$	$(-2,-2,-2,+2)$
$\hat{C}_4$	$+-++$	$(+2,+2,-2,+2)$
$-\hat{C}_4$	$-+--$	$(-2,-2,+2,-2)$
$\overleftarrow{C}_4$	$-+++$	$(2,-2,-2,-2)$
$-\overleftarrow{C}_4$	$+---$	$(-2,2,2,2)$
$\overleftarrow{\hat{C}}_4$	$-+++$	$(2,-2,-2,-2)$
$-\overleftarrow{\hat{C}}_4$	$++-+$	$(2,-2,2,2)$

Table 9.4: *A comparison of Hadamard transform vectors for Golay pair components of length* 2^2.

C	C^T	HC
C_8	$+++-++-+$	$(4,0,4,0,0,4,0,-4)$
$-C_8$	$---+--+-$	$(-4,0,-4,0,0,-4,0,4)$
$\hat{C}_8$	$+-+++--$	$(0,4,0,4,4,0,-4,0)$
$-\hat{C}_8$	$-+---++$	$(0,-4,0,-4,-4,0,4,0)$
$\overleftarrow{C}_8$	$+-++-++$	$(4,0,-4,0,0,4,0,4)$
$-\overleftarrow{C}_8$	$-+--+---$	$(-4,0,4,0,0,-4,0,-4)$
$\overleftarrow{\hat{C}}_8$	$+++---+-$	$(0,4,0,-4,4,0,4,0)$
$-\overleftarrow{\hat{C}}_8$	$---+++-+$	$(0,-4,0,4,-4,0,-4,0)$

Table 9.5: *A comparison of Hadamard transform vectors for Golay pair components of length* 2^3.

C	C^T and HC
C_{16}	$+++-+-+++++---+-$ $(4,4,4,-4,4,4,4,-4,4,-4,4,4,-4,4,-4,-4)$
$-C_{16}$	$---+--+-----++-+$ $(-4,-4,-4,4,-4,-4,-4,4,-4,4,-4,-4,4,-4,4,4)$
$\hat{C}_{16}$	$+-++++---+-++-++$ $(4,4,-4,4,4,4,-4,4,-4,4,4,4,4,-4,-4,-4)$
$-\hat{C}_{16}$	$-+---++++-+--+--$ $(-4,-4,4,-4,-4,-4,4,-4,4,-4,-4,-4,-4,4,4,4)$
$\overleftarrow{C}_{16}$	$-+---++++-++-++$ $(4,-4,-4,-4,-4,4,4,4,-4,-4,4,-4,-4,-4,4,-4)$
$-\overleftarrow{C}_{16}$	$+-+++----+--+--$ $(-4,4,4,4,4,-4,-4,-4,4,4,-4,4,4,4,-4,4)$
$\overleftarrow{\hat{C}}_{16}$	$+++-+++-+---+++-+$ $(4,-4,4,4,-4,4,-4,-4,4,4,4,-4,4,4,4,-4)$
$-\overleftarrow{\hat{C}}_{16}$	$---+--+-+++---+-$ $(-4,4,-4,-4,4,-4,4,4,-4,-4,-4,4,-4,-4,-4,4)$

Table 9.6: *A comparison of Hadamard transform vectors for Golay pair components of length* 2^4.

C	C^T and HC
C_{32}	+ + + − + + − + + + + − − − + − + + + − + + − + − − − + + + − + (8, 0, 8, 0, 0, 8, 0, -8, 8, 0, 8, 0, 0, 8, 0, -8, 0, 8, 0, -8, 8, 0, 8, 0, 0, -8, 0, 8, -8, 0, -8, 0)
$-C_{32}$	− − − + − − + − − − − + + + − + − − − + − − + − + + + − − − + − (-8, 0, -8, 0, 0, -8, 0, 8, -8, 0, -8, 0, 0, -8, 0, 8, 0, -8, 0, 8, -8, 0, -8, 0, 0, 8, 0, -8, 8, 0, 8, 0)
$\hat{C}_{32}$	+ − + + + − − − + − + + − + + + + − + + + − − − − + − − + − − − (0, 8, 0, 8, 8, 0, -8, 0, 0, 8, 0, 8, 8, 0, -8, 0, 8, 0, -8, 0, 0, 8, 0, 8, -8, 0, 8, 0, 0, -8, 0, -8)
$-\hat{C}_{32}$	− + − − − + + + − + − − + − − − − + − − − + + + + − + + − + ++ (0, -8, 0, -8, -8, 0, 8, 0, 0, -8, 0, -8, -8, 0, 8, 0, -8, 0, 8, 0, 0, -8, 0, -8, 8, 0, -8, 0, 0, 8, 0, 8)
$\overleftarrow{C}_{32}$	+ − + + + − − − + − + + − + + + − + − − − + + + + − + + − + ++ (8, 0, -8, 0, 0, 8, 0, 8, -8, 0, 8, 0, 0, -8, 0, -8, 0, 8, 0, 8, 8, 0, -8, 0, 0, 8, 0, 8, 8, 0, -8, 0)
$-\overleftarrow{C}_{32}$	− + − − − + + + − + − − + − − − + − + + + − − − − + − − + − −− (-8, 0, 8, 0, 0, -8, 0, -8, 8, 0, -8, 0, 0, 8, 0, 8, 0, -8, 0, -8, -8, 0, 8, 0, 0, -8, 0, -8, -8, 0, 8, 0)
$\overleftarrow{\hat{C}}_{32}$	− − − + − − + − − − − + + + − + + + + − + + − + − − − + + + − + (0, -8, 0, 8, -8, 0, -8, 0, 0, 8, 0, -8, 8, 0, 8, 0, -8, 0, -8, 0, 0, -8, 0, 8, -8, 0, -8, 0, 0, -8, 0, 8)
$-\overleftarrow{\hat{C}}_{32}$	+ + + − + + − + + + + − − − + − − − − + − − + − + + + − − − + − (0, 8, 0, -8, 8, 0, 8, 0, 0, -8, 0, 8, -8, 0, -8, 0, 8, 0, 8, 0, 0, 8, 0, -8, 8, 0, 8, 0, 0, 8, 0, -8)

Table 9.7: *A comparison of Hadamard transform vectors for Golay pair components of length* 2^5.

The equations in this section have all so far referred to the six equivalence operations found at the start of chapter 4. These six operations only produce a small number of Golay pairs, as inequivalent Golay pairs cannot be created in this manner. In fact for the cliques of Golay pairs identified in chapter 7, each Golay pair component has a distinct Hadamard transform when a Sylvester Hadamard matrix is used. First consider the tables for the clique of size 5 (see tables 9.8 and 9.9). It can be seen that the Hadamard transforms are all distinct.

Tables could be presented for all 384 Golay pair components. However, note that if three of the operations are considered:

1. The operation "changing the sign of the first" only changes the sign of the Hadamard transform.

2. The operation "changing the sign of the second" only changes the sign of the Hadamard transform.

3. The operation "interchanging the sequences in a pair" only requires interchanging the C and S tables.

C^T and HC
+ + − + + − − − − − − + − + −−
(-4, -4, 4, 4, 4, -4, -4, 4, 4, 4, 4, 4, 4, -4, 4, -4)
+ + + + + − − + − + − + − − ++
(4, -4, -4, 4, 4, -4, 4, -4, 4, 4, 4, 4, 4, 4, -4, -4)
+ + + − − − − + + − + + + − ++
(4, 4, -4, 4, 4, 4, 4, -4, -4, -4, 4, -4, 4, 4, 4, -4)
+ + − + − − + − − + + + − + ++
(4, -4, -4, -4, 4, -4, 4, 4, -4, 4, 4, 4, 4, -4, 4, 4)
+ + + − − − + − − − − + − − +−
(-4, 4, -4, -4, 4, -4, 4, 4, 4, 4, 4, -4, 4, 4, 4, -4)

Table 9.8: *A comparison of Hadamard transform vectors for 5 Golay pair (C) components of length* 2^4 *created from the double overlap investigation in chapter 7.*

S^T and HS
− − + − − + + + − − − + − + −−
(-4, -4, -4, -4, -4, 4, -4, 4, 4, 4, -4, -4, -4, 4, 4, -4)
− − − − − + + − − + − + − − ++
(-4, -4, -4, -4, -4, -4, 4, 4, -4, 4, 4, -4, -4, 4, -4, 4)
− − − + − − − + − + − − + − ++
(-4, -4, -4, 4, -4, -4, 4, -4, -4, -4, -4, 4, 4, 4, -4, 4)
+ + + − − − − + − + − − − + −−
(-4, -4, 4, -4, 4, 4, 4, -4, 4, 4, -4, 4, 4, 4, 4, -4)
− + − − + − − − + − + + + − −−
(-4, 4, 4, 4, 4, -4, -4, -4, -4, -4, 4, -4, -4, -4, 4, -4)

Table 9.9: *A comparison of Hadamard transform vectors for 5 Golay pair (S) components of length* 2^4 *created from the double overlap investigation in chapter 7.*

Thus it is only necessary to present the transform for 48 of the 384 Golay pairs (see tables 9.10 and 9.11).

C	C^T and HC
C_1	+ − + + + − − − − − − + − − +− (-4,4,-4,4,4,-4,-4,4,4,4,4,4,4,4,-4,-4)
C_2	+ − − − + − + + − − + − − − −+ (-4,4,-4,4,-4,4,4,-4,4,4,4,4,-4,-4,4,4)
C_3	+ − + + − + + + − − − + + + −+ (4,-4,-4,4,-4,4,-4,4,4,4,4,-4,-4,4,4,4,4)
C_4	+ − − − − + − − − − + − + + +− (-4,4,4,-4,-4,4,4,-4,4,-4,-4,4,4,4,4,4)
C_5	+ − + + + + + − − + + + − − +− (4,4,-4,-4,4,-4,-4,4,4,4,4,4,4,-4,4,-4,4)
C_6	+ − − − + + − + − + − − − − −+ (-4,-4,4,4,-4,4,4,4,-4,4,4,4,4,4,-4,4,-4,4)
C_7	+ − + + − − − + − + + + + + −+ (4,-4,-4,4,4,4,4,-4,-4,-4,4,-4,4,4,4,4,4)
C_8	+ − − − − − − + − − + − − + + +− (-4,4,4,4,-4,-4,-4,4,4,4,-4,4,4,-4,4,4,4,4)
C_9	+ − + − + + − − − − − − − − + +− (-4,4,4,4,-4,-4,4,4,-4,4,4,4,4,4,4,4,4,-4,-4)
C_{10}	+ − − + + + + + − − + + − + −+ (4,-4,-4,4,4,-4,4,4,-4,4,4,4,4,4,-4,-4,4,4)
C_{11}	+ − + − − − + + − − − − + − −+ (-4,4,-4,4,4,-4,4,4,4,-4,4,4,4,-4,-4,4,4,4)
C_{12}	+ − − + − − − − − − + + + − +− (-4,4,-4,4,4,4,-4,-4,4,4,-4,-4,4,4,4,4,4)
C_{13}	+ − + − − − − − + + − − − + +− (-4,4,4,-4,4,4,4,4,4,-4,4,-4,4,4,4,4,-4,-4)
C_{14}	+ − − + − − + + + + + + − + −+ (4,-4,-4,4,4,4,4,4,4,-4,4,4,-4,4,-4,-4,4,4)
C_{15}	+ − + − − − − − − − + + + − −+ (-4,4,-4,4,4,4,4,-4,-4,-4,4,4,4,-4,4,4,4,4)
C_{16}	+ − − + − − + + − − − − + − +− (-4,4,-4,4,4,-4,-4,4,4,4,4,-4,-4,4,4,4,4,4)
C_{17}	+ − + + + + + − − + − − + + +− (4,4,4,-4,-4,-4,-4,4,4,4,4,-4,4,4,4,4,-4,4)
C_{18}	+ − − − + + − + − + + + + + −+ (4,-4,4,4,-4,4,4,-4,-4,-4,4,4,4,-4,4,4,4,4)
C_{19}	+ − + + − − − + − + − − − − −+ (-4,-4,-4,4,4,4,4,4,-4,4,4,4,-4,4,4,4,-4,4)
C_{20}	+ − − − − − + − − + + + − − +− (-4,4,4,-4,-4,4,4,-4,4,4,4,4,-4,4,4,4,4,4)
C_{21}	+ − + + − + − − + + + − + + +− (4,4,4,-4,4,4,4,-4,4,-4,-4,-4,4,4,4,4,-4,4)

C_{22}	$+--------+++++-+++-+$ (4,-4,4,4,-4,4,4,4,-4,4,-4,-4,-4,4,4,4)
C_{23}	$+-+-+-+-----+---+$ (-4,-4,-4,4,4,4,-4,4,4,4,4,-4,4,4,-4,4)
C_{24}	$+---+++--+---+-$ (-4,4,-4,-4,-4,4,4,4,4,-4,4,4,-4,4,4,4)
C_{25}	$+-+-+++--+---+--$ (-4,4,4,-4,4,4,-4,-4,4,4,4,4,-4,4,-4,4)
C_{26}	$+--+++++-+-+--++$ (4,-4,-4,4,-4,-4,4,4,4,4,4,4,-4,4,-4,4)
C_{27}	$+-+---++-+++-+++$ (4,4,-4,-4,-4,4,4,-4,-4,4,-4,4,4,4,4,4)
C_{28}	$+--+--------+-+++--$ (-4,-4,4,4,4,-4,-4,4,-4,4,-4,4,4,4,4,4)
C_{29}	$+--++++--+-+----$ (-4,4,4,4,4,4,-4,4,4,-4,4,4,-4,-4,-4,4)
C_{30}	$+--+--+++-+-+++$ (4,4,-4,4,-4,4,4,4,-4,-4,-4,4,4,-4,4,4)
C_{31}	$+--++++---+-+++$ (4,-4,4,4,-4,-4,-4,4,-4,4,4,4,4,4,-4,4)
C_{32}	$+--+--++-+-+----$ (-4,-4,-4,4,4,-4,4,4,4,4,-4,4,-4,4,4,4)
C_{33}	$+-++++---+--+---$ (-4,4,4,4,4,-4,-4,-4,4,4,-4,4,4,4,-4,4)
C_{34}	$++-++++-+++---+$ (4,-4,4,4,4,-4,4,4,4,4,4,-4,-4,-4,-4,4)
C_{35}	$+++---+-+++-++-+$ (4,4,4,-4,4,4,4,-4,-4,4,-4,-4,4,-4,4,4)
C_{36}	$++-+---+++-+++++-$ (4,-4,4,4,4,-4,4,4,-4,-4,-4,4,4,4,4,-4)
C_{37}	$+++-+++-+++-+--+-$ (4,4,4,-4,4,-4,4,4,4,4,4,-4,-4,4,-4,-4)
C_{38}	$++-+++-++++----+$ (4,-4,4,4,4,4,4,4,-4,4,-4,4,4,-4,-4,-4)
C_{39}	$+++-+++---+-+++-+$ (4,4,4,-4,-4,4,-4,-4,4,4,4,-4,4,-4,4,4)
C_{40}	$++-+++-+---+++++-$ (4,-4,4,4,-4,-4,-4,4,4,-4,4,4,4,4,4,-4)
C_{41}	$+++++++--+-+--++-$ (4,4,4,-4,4,4,4,-4,4,4,4,4,-4,-4,-4,-4)
C_{42}	$+++++--+++-+-+--+$ (4,4,-4,4,4,4,4,4,-4,4,-4,-4,-4,4,-4,4,4)

C_{43}	$+++++++---+-++--+$ (4,-4,4,4,4,-4,-4,-4,4,4,4,-4,4,4,-4,4)
C_{44}	$+++++--++-+-+-+-++-$ (4,-4,-4,-4,4,-4,4,4,4,4,4,-4,4,4,4,4,-4)
C_{45}	$+++++++--+--++-+-$ (4,4,4,4,4,-4,-4,4,4,-4,4,-4,4,4,-4,-4)
C_{46}	$++--+++++++-+--+--$ (4,4,4,4,-4,4,4,4,-4,4,-4,4,-4,-4,-4,4,4)
C_{47}	$+++++++---++--+-+$ (4,-4,4,-4,4,4,4,-4,-4,4,4,4,4,4,4,-4,-4,4)
C_{48}	$++--++++-+-+-++-$ (4,-4,4,-4,-4,-4,4,4,4,4,4,4,4,-4,4,4,-4)

Table 9.10: *A comparison of Hadamard transform vectors for 48 Golay pair (C) components of length* 2^4.

S	S^T and HS
S_1	$-+---++++---+--+-$ (-4,-4,-4,-4,-4,-4,4,4,4,-4,4,-4,-4,4,4,-4)
S_2	$-+++-+-----+----+$ (-4,-4,-4,-4,4,4,4,-4,-4,4,-4,4,4,-4,-4,4)
S_3	$-+--+-------++++-+$ (-4,-4,4,4,-4,-4,-4,-4,-4,4,4,4,-4,4,4,-4)
S_4	$-+++++-++--+-++++-$ (4,4,-4,-4,-4,-4,-4,-4,4,4,-4,-4,4,4,4,-4,-4)
S_5	$-+-----+-+++--+-$ (-4,-4,-4,-4,4,-4,4,4,-4,-4,-4,4,4,-4,4,4,-4)
S_6	$-+++--+--+-----+$ (-4,-4,-4,-4,4,-4,4,-4,4,4,-4,-4,4,-4,-4,4)
S_7	$-+--++--++++++-+$ (4,-4,4,-4,-4,-4,-4,-4,-4,4,4,4,-4,-4,-4,4,4)
S_8	$-+++++-+-+--++++-$ (4,-4,4,-4,-4,-4,-4,-4,4,4,-4,-4,4,4,4,-4,-4)
S_9	$-+-+--++-------++-$ (-4,-4,-4,-4,-4,-4,4,4,4,4,-4,-4,4,4,-4,4,-4)
S_{10}	$-++---------++-+-+$ (-4,-4,-4,-4,4,4,4,-4,-4,-4,4,4,-4,4,-4,4,-4)
S_{11}	$-+-++++-------+--+$ (-4,-4,4,4,4,-4,-4,-4,-4,4,4,-4,4,4,-4,-4,4)

S_{12}	$-++-+++--++-+-$ (4,4,-4,-4,-4,-4,-4,-4,4,-4,4,-4,-4,4,4,-4)
S_{13}	$-+-+---------++-++-$ (-4,-4,-4,-4,4,-4,-4,4,-4,-4,4,4,4,-4,4,-4)
S_{14}	$-++---++-------+-+$ (-4,-4,-4,-4,-4,4,4,-4,4,4,-4,-4,4,-4,4,-4)
S_{15}	$-+-+---+++--+--+$ (-4,-4,4,4,4,-4,4,-4,-4,-4,-4,-4,4,4,-4,-4,4)
S_{16}	$-++---++++++++-+-$ (4,4,-4,-4,4,-4,4,-4,-4,-4,-4,-4,-4,4,4,4,-4)
S_{17}	$-+---------+-+--++-$ (-4,-4,4,-4,-4,-4,4,-4,-4,-4,-4,4,4,4,4,4,-4)
S_{18}	$-+++--+---+++++-+$ (4,-4,-4,-4,4,-4,-4,-4,-4,4,4,-4,-4,4,-4,4,4)
S_{19}	$-+--++++--+-----+$ (-4,-4,4,-4,-4,-4,4,-4,4,4,4,4,-4,-4,-4,-4,4)
S_{20}	$-+++++-+-+++--+-$ (4,-4,-4,-4,4,4,-4,-4,-4,4,-4,4,4,4,-4,4,-4,-4)
S_{21}	$-+---+-------++++-$ (-4,-4,4,-4,-4,-4,-4,4,-4,-4,4,-4,4,4,4,4,-4)
S_{22}	$-+++-+++--+-++-+$ (4,-4,-4,-4,-4,4,4,-4,-4,4,-4,-4,-4,4,-4,4,4,4)
S_{23}	$-+---+--++++---+$ (-4,-4,4,-4,4,4,4,-4,-4,-4,4,-4,-4,-4,-4,4)
S_{24}	$-+++-+++++-+--+-$ (4,-4,-4,-4,4,-4,4,4,4,4,-4,-4,-4,-4,4,-4,-4)
S_{25}	$-+-+--++-+++----$ (-4,-4,-4,-4,4,-4,4,-4,4,-4,-4,4,-4,-4,4,4)
S_{26}	$-++-------+-+--++$ (-4,-4,-4,-4,4,-4,4,-4,-4,4,4,-4,4,4,-4,-4)
S_{27}	$-+-++---++-+++$ (4,-4,4,-4,-4,-4,-4,-4,-4,-4,4,4,4,-4,-4,4)
S_{28}	$-++-+++-+-+++--$ (4,-4,4,-4,-4,-4,-4,-4,4,4,-4,-4,-4,4,4,-4)
S_{29}	$-+-+-----+++--$ (-4,-4,4,-4,4,-4,-4,-4,-4,-4,-4,4,4,-4,4,4)
S_{30}	$-+-+++++-+++---++$ (4,-4,-4,-4,-4,-4,4,4,-4,4,4,4,-4,-4,-4,4)
S_{31}	$-+-+----+--+--++$ (-4,-4,-4,4,4,-4,4,4,-4,-4,4,-4,4,-4,-4,-4)
S_{32}	$-+-++++++--++--$ (4,-4,4,4,-4,-4,-4,4,4,-4,-4,-4,-4,4,4,-4)

S_{33}	$-+---+++-+--+---$ (-4,-4,4,-4,-4,-4,4,-4,4,-4,-4,-4,-4,4,4,4)
S_{34}	$++-+++---+-++-+$ (4,4,4,-4,-4,-4,-4,4,4,-4,4,4,4,-4,4,4)
S_{35}	$+++---+----+--+-$ (-4,4,-4,-4,4,-4,4,4,4,4,4,4,-4,4,4,4,-4)
S_{36}	$++-+---+--+----+$ (-4,-4,-4,4,4,4,4,-4,4,-4,4,4,4,-4,4,4)
S_{37}	$+++---+++-+++-+$ (4,-4,4,4,4,4,4,-4,-4,4,-4,-4,4,4,4,-4)
S_{38}	$++-+--+-+++-++-+$ (4,4,4,-4,4,-4,4,4,-4,-4,-4,4,4,-4,4,4)
S_{39}	$+++----+--+---+-$ (-4,4,-4,-4,4,4,4,4,-4,4,-4,4,4,4,4,4,-4)
S_{40}	$++-+--+----+---+$ (-4,-4,-4,4,4,-4,4,4,4,4,4,-4,4,-4,4,4)
S_{41}	$++--+++++--+-+-+$ (4,-4,4,4,-4,4,4,4,4,4,4,-4,-4,-4,4,-4)
S_{42}	$++-------+--++-+-$ (-4,4,4,4,4,-4,4,4,-4,-4,4,-4,4,4,4,-4)
S_{43}	$++--+++++-+++-+-+-$ (4,4,4,-4,-4,-4,4,-4,4,-4,4,4,-4,4,4,4)
S_{44}	$++-------++--+-+$ (-4,-4,4,-4,4,4,4,4,-4,-4,4,4,4,4,4,-4,4,4)
S_{45}	$+++-+--+++--+-+-+$ (4,-4,-4,4,4,4,4,4,4,4,4,-4,-4,4,-4,4,-4)
S_{46}	$++-------+-+--++-$ (-4,4,4,-4,4,4,4,4,4,-4,-4,4,4,4,4,-4,4,-4)
S_{47}	$+++++--++-++-+-+-$ (4,4,-4,-4,4,4,-4,4,-4,4,-4,-4,4,4,4,4,4,4)
S_{48}	$++-------+-+++--+$ (-4,-4,4,4,4,4,-4,4,4,-4,-4,4,4,-4,4,4,4,4)

Table 9.11: *A comparison of Hadamard transform vectors for 48 Golay pair (S) components of length* 2^4.

Close examination of these tables shows that if the Hadamard transform is precisely aligned with the C or S component, then the C or S component can be identified to within a plus or minus sign. Note that the fact that the transforms are unique has been proved here for all Golay pairs obtained by the doubling construction. The uniqueness results observed for pairs of length 16 not obtained by the doubling construction would need to be checked for other specific lengths.

It is now necessary to consider what happens if the components are not precisely aligned.

In table 9.1 the eighth row arises from transforming 8 bits of external padding concatenated with the first 8 bits of C or S. Denote the vectors $(\mathbf{0}_n C_n)$ and $(\mathbf{0}_n S_n)$ (of length $2n$) by $\tilde{C}$ and $\tilde{S}$ respectively. Using the fact that the first half of C_n or S_n is simply $C_{n/2}$, the two transforms can be calculated as follows:

$$H_2\tilde{C}_2 = H_2\tilde{S}_2 = \begin{pmatrix} C_1 \\ -C_1 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

$$H_4\tilde{C}_4 = H_4\tilde{S}_4 = \begin{pmatrix} C_1 + S_1 \\ C_1 - S_1 \\ -C_1 - S_1 \\ -C_1 + S_1 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \\ -2 \\ 0 \end{pmatrix}$$

$$H_8\tilde{C}_8 = H_8\tilde{S}_8 = \begin{pmatrix} 2C_1 \\ 2C_1 \\ 2S_1 \\ -2S_1 \\ -2C_1 \\ -2C_1 \\ -2S_1 \\ 2S_1 \end{pmatrix} = \begin{pmatrix} 2 \\ 2 \\ 2 \\ -2 \\ -2 \\ -2 \\ -2 \\ 2 \end{pmatrix}$$

$$H_{16}\tilde{C}_{16} = H_{16}\tilde{S}_{16} = \begin{pmatrix} 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ -2C_1 + 2S_1 \\ -2C_1 - 2S_1 \\ -2C_1 - 2S_1 \\ -2C_1 + 2S_1 \\ -2C_1 - 2S_1 \\ -2C_1 + 2S_1 \\ -2C_1 + 2S_1 \\ -2C_1 - 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \end{pmatrix} = \begin{pmatrix} 4 \\ 0 \\ 4 \\ 0 \\ 0 \\ 4 \\ 0 \\ -4 \\ -4 \\ 0 \\ -4 \\ 0 \\ 0 \\ -4 \\ 0 \\ 4 \end{pmatrix}$$

$$H_{32}\tilde{C}_{32} = H_{32}\tilde{S}_{32} = \begin{pmatrix} 4C_1 \\ 4C_1 \\ 4S_1 \\ -4S_1 \\ 4C_1 \\ 4C_1 \\ 4S_1 \\ -4S_1 \\ 4S_1 \\ -4S_1 \\ 4C_1 \\ 4C_1 \\ -4S_1 \\ 4S_1 \\ -4C_1 \\ -4C_1 \\ -4C_1 \\ -4C_1 \\ -4S_1 \\ 4S_1 \\ -4C_1 \\ -4C_1 \\ -4S_1 \\ 4S_1 \\ -4S_1 \\ 4S_1 \\ -4C_1 \\ -4C_1 \\ 4S_1 \\ -4S_1 \\ 4C_1 \\ 4C_1 \end{pmatrix} = \begin{pmatrix} 4 \\ 4 \\ 4 \\ -4 \\ 4 \\ 4 \\ 4 \\ -4 \\ 4 \\ -4 \\ 4 \\ 4 \\ -4 \\ 4 \\ -4 \\ -4 \\ -4 \\ -4 \\ -4 \\ 4 \\ -4 \\ -4 \\ -4 \\ 4 \\ -4 \\ 4 \\ -4 \\ -4 \\ 4 \\ -4 \\ 4 \\ 4 \end{pmatrix}$$

If we consider $n > 1$ (where n is a power of 2), it is possible to state this generally, as shown in equation 9.15:

$$H_n\tilde{C}_n = \begin{pmatrix} H_{\frac{n}{2}}C_{\frac{n}{2}} \\ -H_{\frac{n}{2}}C_{\frac{n}{2}} \end{pmatrix} \qquad H_n\tilde{S}_n = \begin{pmatrix} H_{\frac{n}{2}}C_{\frac{n}{2}} \\ -H_{\frac{n}{2}}C_{\frac{n}{2}} \end{pmatrix} \tag{9.15}$$

The new Hadamard transform vector is the concatenation of two smaller transforms. It is possible for a third party identifying the transform to see that the second half of the transform is simply minus the first half. This implies that the Golay pair is not yet identified, and that it is necessary to wait until the

vector in table 9.1 is found before it is possible to identify the Golay pair (as shown earlier in this chapter).

Another possibility can be dealt with theoretically. Consider the vector arising from the last half of C or S concatenated with a second half of external padding. Denote the vectors $(S_n \mathbf{0}_n)$ and $(-S_n \mathbf{0}_n)$ (of length $2n$) by $\breve{C}$ and $\breve{S}$ respectively. Using the fact that the second halves of C_n and S_n are $S_{n/2}$ and $-S_{n/2}$ respectively, the two transforms can be calculated as follows:

$$H_2\breve{C}_2 = \begin{pmatrix} S_1 \\ S_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \qquad H_2\breve{S}_2 = \begin{pmatrix} -S_1 \\ -S_1 \end{pmatrix} = \begin{pmatrix} -1 \\ -1 \end{pmatrix}$$

$$H_4\breve{C}_4 = \begin{pmatrix} C_1 - S_1 \\ C_1 + S_1 \\ C_1 - S_1 \\ C_1 + S_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 2 \\ 0 \\ 2 \end{pmatrix} \qquad H_4\breve{S}_4 = \begin{pmatrix} -C_1 + S_1 \\ -C_1 - S_1 \\ -C_1 + S_1 \\ -C_1 - S_1 \end{pmatrix} = \begin{pmatrix} 0 \\ -2 \\ 0 \\ -2 \end{pmatrix}$$

$$H_8\breve{C}_8 = \begin{pmatrix} 2S_1 \\ -2S_1 \\ 2C_1 \\ 2C_1 \\ 2S_1 \\ -2S_1 \\ 2C_1 \\ 2C_1 \end{pmatrix} = \begin{pmatrix} 2 \\ -2 \\ 2 \\ 2 \\ 2 \\ -2 \\ 2 \\ 2 \end{pmatrix} \qquad H_8\breve{S}_8 = \begin{pmatrix} -2S_1 \\ 2S_1 \\ -2C_1 \\ -2C_1 \\ -2S_1 \\ 2S_1 \\ -2C_1 \\ -2C_1 \end{pmatrix} = \begin{pmatrix} -2 \\ 2 \\ -2 \\ -2 \\ -2 \\ 2 \\ -2 \\ -2 \end{pmatrix}$$

$$H_{16}\breve{C}_{16} = \begin{pmatrix} 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ -2C_1 + 2S_1 \\ -2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ -2C_1 + 2S_1 \\ -2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ 2C_1 - 2S_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 4 \\ 0 \\ -4 \\ 4 \\ 0 \\ 4 \\ 0 \\ 0 \\ 4 \\ 0 \\ -4 \\ 4 \\ 0 \\ 4 \\ 0 \end{pmatrix} \qquad H_{16}\breve{S}_{16} = \begin{pmatrix} -2C_1 + 2S_1 \\ -2C_1 - 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ -2C_1 - 2S_1 \\ -2C_1 + 2S_1 \\ -2C_1 - 2S_1 \\ -2C_1 + 2S_1 \\ -2C_1 + 2S_1 \\ -2C_1 - 2S_1 \\ 2C_1 - 2S_1 \\ 2C_1 + 2S_1 \\ -2C_1 - 2S_1 \\ -2C_1 + 2S_1 \\ -2C_1 - 2S_1 \\ -2C_1 + 2S_1 \end{pmatrix} = \begin{pmatrix} 0 \\ -4 \\ 0 \\ 4 \\ -4 \\ 0 \\ -4 \\ 0 \\ 0 \\ -4 \\ 0 \\ 4 \\ -4 \\ 0 \\ -4 \\ 0 \end{pmatrix}$$

$$H_{32}\breve{C}_{32} = \begin{pmatrix} 4S_1 \\ -4S_1 \\ 4C_1 \\ 4C_1 \\ -4S_1 \\ 4S_1 \\ -4C_1 \\ -4C_1 \\ 4C_1 \\ 4C_1 \\ 4S_1 \\ -4S_1 \\ 4C_1 \\ 4C_1 \\ 4S_1 \\ -4S_1 \\ 4S_1 \\ -4S_1 \\ 4C_1 \\ 4C_1 \\ -4S_1 \\ 4S_1 \\ -4C_1 \\ -4C_1 \\ 4C_1 \\ 4C_1 \\ 4S_1 \\ -4S_1 \\ 4C_1 \\ 4C_1 \\ 4S_1 \\ -4S_1 \end{pmatrix} = \begin{pmatrix} 4 \\ -4 \\ 4 \\ 4 \\ -4 \\ 4 \\ -4 \\ -4 \\ 4 \\ 4 \\ 4 \\ -4 \\ 4 \\ 4 \\ 4 \\ -4 \\ 4 \\ -4 \\ 4 \\ 4 \\ -4 \\ 4 \\ -4 \\ -4 \\ 4 \\ 4 \\ 4 \\ -4 \\ 4 \\ 4 \\ 4 \\ -4 \end{pmatrix} \qquad H_{32}\breve{S}_{32} = \begin{pmatrix} -4S_1 \\ 4S_1 \\ -4C_1 \\ -4C_1 \\ 4S_1 \\ -4S_1 \\ 4C_1 \\ 4C_1 \\ -4C_1 \\ -4C_1 \\ -4S_1 \\ 4S_1 \\ -4C_1 \\ -4C_1 \\ -4S_1 \\ 4S_1 \\ -4S_1 \\ 4S_1 \\ -4C_1 \\ -4C_1 \\ 4S_1 \\ -4S_1 \\ 4C_1 \\ 4C_1 \\ -4C_1 \\ -4C_1 \\ -4S_1 \\ 4S_1 \\ -4C_1 \\ -4C_1 \\ -4S_1 \\ 4S_1 \end{pmatrix} = \begin{pmatrix} -4 \\ 4 \\ -4 \\ -4 \\ 4 \\ -4 \\ 4 \\ 4 \\ -4 \\ -4 \\ -4 \\ 4 \\ -4 \\ -4 \\ -4 \\ 4 \\ -4 \\ 4 \\ -4 \\ -4 \\ 4 \\ -4 \\ 4 \\ 4 \\ -4 \\ -4 \\ -4 \\ 4 \\ -4 \\ -4 \\ -4 \\ 4 \end{pmatrix}$$

If we consider $n > 1$ (where n is a power of 2), it is possible to state this generally, as shown in equation 9.16.

$$H_n\tilde{C}_n = \begin{pmatrix} H_{\frac{n}{2}}S_{\frac{n}{2}} \\ H_{\frac{n}{2}}S_{\frac{n}{2}} \end{pmatrix} \qquad H_n\tilde{S}_n = \begin{pmatrix} -H_{\frac{n}{2}}S_{\frac{n}{2}} \\ -H_{\frac{n}{2}}S_{\frac{n}{2}} \end{pmatrix} \tag{9.16}$$

1	-1	-1	1	-1	1	1	-1	-1	1	1	-1	1	-1	-1	1
0	2	0	-2	0	-2	0	2	0	-2	0	2	0	2	0	-2
1	-3	1	1	-1	3	-1	-1	-1	3	-1	-1	1	-3	1	1
2	2	-2	2	-2	-2	2	-2	-2	-2	2	-2	2	2	-2	2
3	-3	-3	-1	-1	1	1	3	-3	3	3	1	1	-1	-1	-3
2	4	2	-4	-2	0	-2	0	-2	-4	-2	4	2	0	2	0
1	-3	5	1	1	-3	-3	1	-1	3	-5	-1	-1	3	3	-1
0	**4**	**0**	**4**	**4**	**0**	**-4**	**0**	**0**	**-4**	**0**	**-4**	**-4**	**0**	**4**	**0**
-1	-3	-3	-1	5	-1	-1	-3	3	1	1	3	-7	3	3	1
-2	4	2	-4	6	0	2	0	2	0	-2	0	-6	-4	-2	4
-3	-3	5	1	3	3	3	-1	5	-3	-3	1	-5	3	-5	-1
-2	2	-2	6	-2	2	6	-2	6	2	-2	-2	-2	-6	-2	-2
-3	-1	-5	-3	-3	-1	3	5	9	-5	-1	1	1	3	-1	1
-4	2	4	-6	-4	2	-4	2	8	6	0	-2	0	-2	0	-2
-3	-3	5	5	-3	-3	-3	-3	5	-3	5	-3	5	-3	-3	5
-4	**4**	**-4**	**4**	**4**	**-4**	**-4**	**4**	**4**	**4**	**4**	**4**	**4**	**4**	**-4**	**-4**
-5	-3	-3	-5	5	3	-5	-3	1	-1	-1	1	7	-7	1	-1
-4	2	4	-2	4	-2	4	-6	0	2	0	-2	8	6	0	2
-5	-1	3	3	5	1	5	5	-3	1	5	-3	3	-1	3	-5
-6	2	-2	2	-2	6	2	-2	-2	-2	2	6	2	2	6	2
-7	-1	-1	-3	-3	-5	3	1	-5	5	-3	-1	-1	1	1	3
-6	0	2	0	-2	4	-2	4	-6	-4	2	-4	-2	0	-2	0
-5	-1	-1	3	-5	-1	-1	-5	-3	1	1	5	-3	1	1	-3
-4	**0**	**-4**	**0**	**0**	**-4**	**0**	**4**	**-4**	**0**	**-4**	**0**	**0**	**-4**	**0**	**4**
-3	-1	-1	-3	-1	5	-3	-1	-3	-1	-1	-3	-1	5	-3	-1
-2	0	2	0	-2	-4	2	-4	-2	0	2	0	-2	-4	2	-4
-1	-1	-1	3	1	1	1	5	-1	-1	-1	3	1	1	1	5
-2	2	-2	-2	-2	2	-2	-2	-2	2	-2	-2	-2	2	-2	-2
-1	-3	1	-1	-1	-3	1	-1	-1	-3	1	-1	-1	-3	1	-1
0	2	0	2	0	2	0	2	0	2	0	2	0	2	0	2
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1

Table 9.12: *A second example of using a* 16×16 *Hadamard matrix in a Hadamard transform to locate and identify the C component used in an LS code of length 1296.*

Tables 9.1, 9.12 and 9.13 show three examples of three different LS codes created using three different Golay pairs.

The changing transform vector, obtained by shifting through the Golay pair component as in tables 9.1, 9.12 and 9.13, shows the same behaviour for

all 384 Golay pairs. The only transform vectors with entries restricted to 0 and ± 4 are those arising from half overlap or complete overlap of length 16 Golay pair components. This is true whether shifting through a C component or a S component. Clearly space does not allow the presentation of all 384 tables.

1	-1	-1	1	-1	1	1	-1	-1	1	1	-1	1	-1	-1	1
0	2	0	-2	0	-2	0	2	0	-2	0	2	0	2	0	-2
-1	-1	3	-1	1	1	-3	1	1	1	-3	1	-1	-1	3	-1
-2	2	2	2	2	-2	-2	-2	2	-2	-2	-2	-2	2	2	2
-1	-3	-3	3	3	1	1	-1	1	3	3	-3	-3	-1	-1	1
-2	4	-2	-4	2	0	2	0	2	-4	2	4	-2	0	-2	0
-1	-5	3	-1	-1	3	3	-1	1	5	-3	1	1	-3	-3	1
0	**4**	**0**	**4**	**-4**	**0**	**4**	**0**	**0**	**-4**	**0**	**-4**	**4**	**0**	**-4**	**0**
-1	-3	-3	-1	-3	-1	-1	5	3	1	1	3	1	3	3	-7
-2	4	2	-4	-2	0	-6	0	2	0	-2	0	2	-4	6	4
-1	-5	3	3	1	-3	-3	-3	-1	3	3	-5	1	5	-3	5
-2	6	-2	2	6	-2	-2	2	-2	-2	6	2	-2	-2	-2	-6
-3	-5	-1	-3	5	3	-1	-3	1	-1	-5	9	1	-1	3	1
-4	6	4	-2	4	-2	4	-2	0	2	-8	-6	0	2	0	2
-5	-5	3	3	3	3	3	3	3	-5	3	-5	-5	3	3	-5
-4	**4**	**-4**	**4**	**-4**	**4**	**4**	**-4**	**4**	**4**	**4**	**4**	**-4**	**-4**	**4**	**4**
-5	-3	-3	-5	-3	-5	3	5	1	-1	-1	1	-1	1	-7	7
-4	2	4	-2	-4	6	-4	2	0	2	0	-2	0	-2	-8	-6
-3	-3	1	5	-5	-5	-1	-5	3	-5	-1	3	5	-3	1	-3
-2	2	-6	2	2	-2	-2	6	2	6	-2	-2	6	2	2	2
-3	-1	-1	-7	1	3	-5	-3	-1	-3	5	-5	3	1	1	-1
-2	0	6	0	2	-4	2	-4	-2	4	6	4	2	0	2	0
-3	1	1	5	5	1	1	5	-5	-1	-1	3	3	-1	-1	3
-4	**0**	**-4**	**0**	**0**	**4**	**0**	**-4**	**-4**	**0**	**-4**	**0**	**0**	**4**	**0**	**-4**
-3	-1	-1	-3	-1	-3	5	-1	-3	-1	-1	-3	-1	-3	5	-1
-2	0	2	0	-2	4	2	4	-2	0	2	0	-2	4	2	4
-3	1	1	1	-5	-1	-1	-1	-3	1	1	1	-5	-1	-1	-1
-2	-2	-2	2	-2	-2	-2	2	-2	-2	-2	2	-2	-2	-2	2
-1	1	-3	-1	-1	1	-3	-1	-1	1	-3	-1	-1	1	-3	-1
0	-2	0	-2	0	-2	0	-2	0	-2	0	-2	0	-2	0	-2
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 9.13: *A third example of using a* 16×16 *Hadamard matrix in a Hadamard transform to locate and identify the* C *component used in an LS code of length 1296.*

9.3 Identifying the entire codeword

If the Hadamard transform is applied piecewise along the codeword (as seen in tables 9.1, 9.12 and 9.13) then the complete code can be established. Note that if the length of the Golay pair components is not known, the use of different sizes of Hadamard matrix for the transform would identify the length. It is necessary to take care to discover the correct length and not half the length, as shown in section 9.2.

Firstly, identify the first Golay C component. If just the Golay C component was sent, the length should equal the number of shifts taken to reach the Hadamard transform of the first Golay pair component from a zero transform. As LS codes use padding, this number is usually larger and the padding is referred to as external padding. This external padding can be calculated as the number of extra shifts needed to find the first Golay pair component (e.g. if a length 16 Golay pair component was used but 31 shifts are needed to find the first Golay pair component, then the external padding is of length 15).

Secondly, now that the lengths of the Golay pair components are known, the distances between each Hadamard transform that finds a Golay pair component can be calculated. These distances identify the internal paddings used, as each extra shift between two Golay pair components can be seen (e.g. if a length 16 Golay pair component was used and there are 21 shifts between two Golay pair components, then the internal padding between them is of length 5).

Thirdly, as every Hadamard transform of each Golay pair component is unique it is possible using a lookup table to identify the Golay pair component to within a plus or minus sign. As each Hadamard transform of a Golay pair component (whether a C_0, C_1, S_0 or S_1) is received during the shifting, it is possible to identify if the Golay pair component or minus the Golay pair component was sent, relative to the sign of the first sent component of the type. This can repeated along the whole codeword and all of the Golay pair components will be found.

Fourthly, it is possible to identify the central external padding that appears in an LS codeword. Using the method similar to identifying the internal padding, this can be found between the last C_0 (or C_1) and the first S_0 (or S_1). Note that it is normal for the central external padding to be longer than the internal padding.

Finally, the Hadamard matrix row used in the construction of the LS codeword can be determined. It is necessary to assume the sign attached to the first instance of each component C_0, C_1, S_0 or S_1. It is simplest to use the facts

that:

$$C_1(z) = z^{N-1}S_0(z^{-1})$$
$$S_1(z) = -z^{N-1}C_0(z^{-1})$$

to determine the relative signs.

Note that the code could have been constructed using the opposite signs for some or all of these components with the opposite signs in the corresponding positions in the Hadamard matrix row. However this uncertainty is unimportant to the third party.

Suppose, for example, that the first half of the codeword can be identified as:

$$C_0, C_1, C_0, -C_1, -C_0, -C_1, C_0, -C_1$$
$$+ + + - - - +-$$

The signs will be identical during the second half of the codeword:

$$S_0, S_1, S_0, -S_1, -S_0, -S_1, S_0, -S_1$$
$$+ + + - - - +-$$

The work of this section has shown, partly theoretically and partly computationally, that it is possible to identify the entire codeword. It is simply necessary to compute the transform of a window of length N sliding through the codeword. Particular transforms are identified when the window corresponds to particular C or S components. The remaining components of the construction are then easily deduced.

Note As noted in section 9.2 it is possible, as an alternative, to consider a Hadamard transform using an $n \times n$ Hadamard matrix, where n is the length of the code. This may seem a more standard approach, but so far it has not been possible to demonstrate that all components of the LS codeword can be correctly identified.

Chapter 10

Observation and prediction of the code by a third party

In this chapter certain mechanisms described in chapter 8 will be investigated in detail. The way that an observer of codewords (using techniques of chapter 9 or alternative techniques) could determine or predict the construction will be studied in detail. Graphs are presented showing the way that knowledge is gained or predictions become more accurate. The aim is to show how a third party could learn about the codes in use and predict the codes when attempting to intercept and recover information or disrupt transmissions.

In section 10.1 a graph is presented that shows how long it takes to achieve full knowledge of various components of the LS code in use. In section 10.2 further graphs will show how knowledge of the complete sets of components from which the individual components are selected can be built up. In section 10.3 graphs will show how the correlation of the true codeword and a predicted codeword changes as more of the true codeword is read. In section 10.4 the mechanism for evolving internal padding will be refined by the use of a recency list. In sections 10.5 and 10.6 comparisons will be made with m-sequences and Gold codes.

Note that in this chapter the length and other parameters of the LS code will always be assumed to be known. These parameters could be determined by a third party relatively easily and do not change during evolution. In a similar way, the vector π which determines the order in which the Golay pair components are used (see chapter 2) will be assumed to be known. Again it does not change during evolution. In the graphs the π vector will be assumed to begin $(01\ldots)$. This is the worst case in the sense that the third party is able to learn details of Golay pair components C_0, S_0, C_1 and S_1 most quickly.

As noted in chapter 9, the code could have been constructed using the opposite signs for some or all of the components C_0, S_0, C_1 and S_1, together

with the opposite signs in the corresponding positions in the Hadamard matrix row. It was noted that this uncertainty is unimportant to the third party. Thus it is convenient to assume that the first two positions of the Hadamard matrix row are +1. Similarly, it will be assumed that the relationship between a Golay pair and a Golay pair mate is $C_1(z) = z^{N-1}S_0(z^{-1})$, $S_1(z) = -z^{N-1}C_0(z^{-1})$ and not the alternative possibility $C_1(z) = -z^{N-1}S_0(z^{-1})$, $S_1(z) = +z^{N-1}C_0(z^{-1})$. If either of these assumptions is false an equivalent Hadamard matrix to the one used to construct the code is found, but the same codeword is reconstructed. In the graphs, the unit of time corresponds to the length of a single LS codeword.

10.1 Identification of components currently in use

In figure 10.1 knowledge has a simple raw definition. Full knowledge of the Hadamard row and the internal padding both contribute 0.25 to the measure of knowledge (i.e Hadamard row: 0.25 and internal padding: 0.25). Full knowledge of the Golay pair except for the signs of S_0 and C_1 contribute 0.25 to the measure of knowledge (i.e C_0 and S_1: 0.25). Knowledge of the actual signs of the S_0 and C_1 components also contribute 0.25 to the measure of knowledge (i.e the sign of S_0 and C_1: 0.25).

Within any one generation a single Hadamard matrix, a single Golay pair and mate, and a single choice of internal padding is used, with a single row of the Hadamard matrix selected for each user. It is possible within this single generation to find out which components are used by applying methods described in chapter 9 over the complete duration of a single bit (i.e. a complete codeword). However, individual components may be identified earlier. As already noted, in this identification only the signs (in the notation of equation 2.7 of chapter 2) of $h_{k,1}C_{\pi_1}$ $h_{k,j}C_{\pi_j}$ are known (where π_j is the first nonzero position in the vector π). The individual signs of $h_{k,1}$, C_{π_1} etc. are not known. However, this is unimportant as the same code is obtained.

For example, assume that a third party identifies a codeword from the beginning and that they want to find out the components of the LS code in use. It is possible to establish the time to identify the components. It was established in chapter 9 that the Golay pair component C_0 can be found soon after the initial external padding is received. At this point S_1 is known from our assumption. The possibilities for S_0 and C_1 are already very limited. These possibilities are further limited as soon as C_1 is found (to within a $\pm$ sign). The

Hadamard matrix row in use is the same in the two parts of the codeword, so S_0, C_1 can be confirmed soon after the external padding in the middle of the LS codeword is received. In the graphs presented here, the standard example from chapter 2 of a code of length 1296 using a 32×32 Hadamard matrix and Golay pair components of length 16 is used. Suppose that C_0 and C_1 are known. By observing the signs multiplying C_0 and C_1 it is possible after 20 positions to determine the Hadamard matrix row used in the construction. After the first 20 positions the Hadamard row is unique in this example. This is the worst case of all the Hadamard matrix rows identified in chapter 5. However, the sign of C_1 is not known until the first S_0 component is read. The internal padding can be found by observing the zeros between the Golay pair components. The internal padding vector is certainly known once half the LS codeword is received. It may be known earlier if the full set of internal padding vectors in use is already known. This initial graph can be seen in figure 10.1.

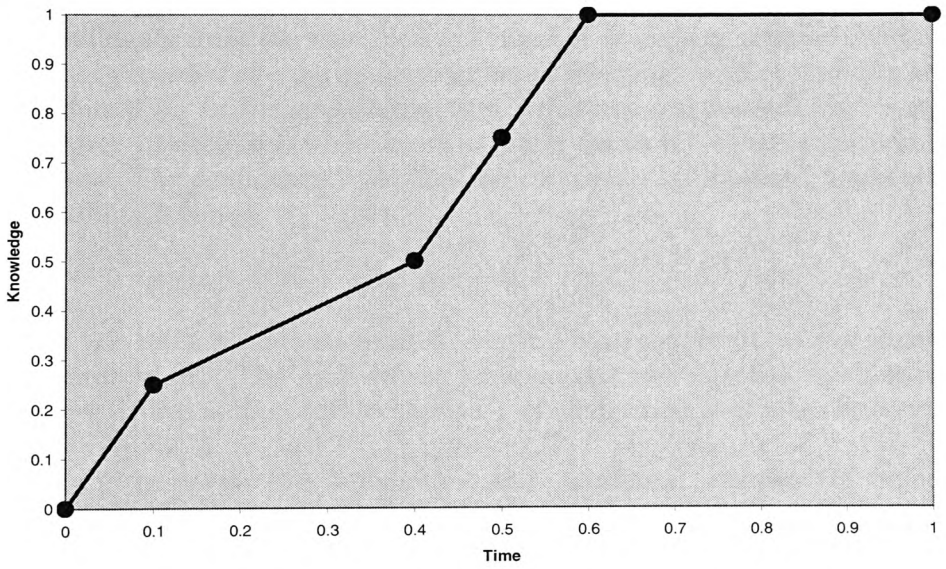

Figure 10.1: *Knowledge gain during a single LS codeword. The four stages of gaining knowledge represent (i) C_0 and S_1, (ii) the internal padding vector, (iii) the Hadamard matrix row, and finally (iv) the sign of S_0 and C_1. The points are interpolated linearly.*

10.2 Identification of all components in use during the evolution of the code

In this section the rate at which knowledge of the complete set of components in use throughout the evolution of the code can be built up by a third party observer will be investigated. First this will be done for each of the components individually. Then it will be done for all of the components together. As a typical illustrative example, the following sets of components are used in this section: 5 Golay pairs, 928 internal padding vectors and 1024 Hadamard matrix rows. Graphs are presented for the cases where components are selected for successive evolutions of the code with repetition. Simulations simply select components randomly with repetition and note the number of distinct components that have been observed after a given number of selections. Results of simulations are presented, together with 95%, 99% and 100% confidence limits for the maximum number of components that could be observed after a given number of selections. A 95% confidence limit (for example) is a number of distinct selected components which is not exceeded at a particular number of selections with probability at least 0.95. Define $p_j(i)$ as the probability that i distinct components have not been selected after j selections (with repetition) and define $|C|$ as the total number of components. The confidence limits can be computed by repeated application of the probability formula

$$p_j(i) = \left((|C| - i)p_{j-1}(i) + (i+1)p_{j-1}(i+1)\right)/|C|.$$

Note that the 100% confidence limit is identical with the results of any simulation without repetition. The case where components are selected randomly with repetition (corresponding, to Mechanism 1 of chapter 8) is of most interest.

Figure 10.2 presents the maximum and minimum number of selections needed to find the five distinct Golay pair components in ten simulations, together with 95%, 99% and 100% confidence limits for the maximum number of components that could be observed after a given number of selections. As noted above, the 100% confidence limits also give the "without repetition" case. In this particular graph there is no difference between the 100% and 99% cases, but at the 95% confidence interval it takes one more generation to find all of the components used.

Figure 10.3 shows a comparison between 100% (with or without repetition), 99% and 95% (with repetition) confidence limits for the 928 internal padding components.

Figure 10.4 shows ten simulations of the number of selections to find all 928 internal padding components used in the construction.

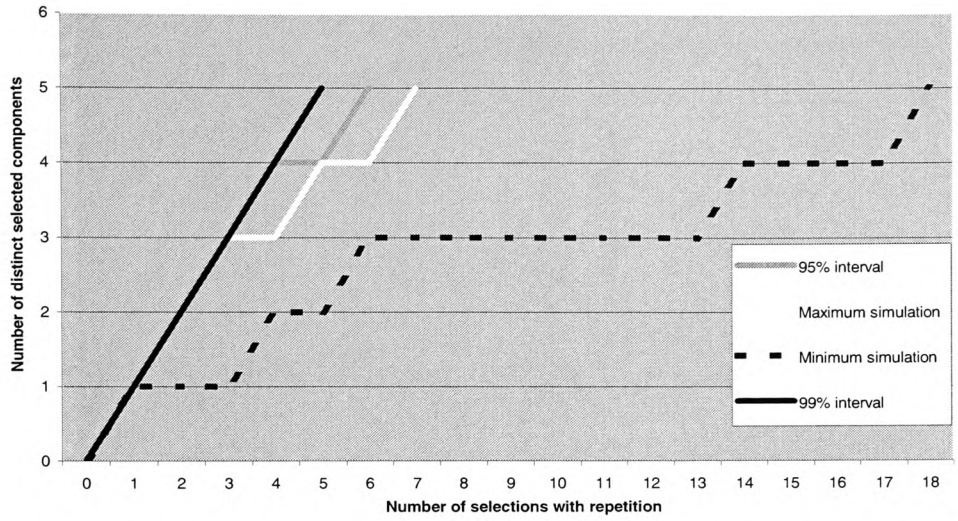

Figure 10.2: *The maximum and minimum of ten simulations for the number of distinct pairs observed out of five Golay pairs. Selection of the next Golay pair is made with repetition allowed. Also shown are* 95% *and* 99% *confidence limits for the maximum number of pairs that could be observed.*

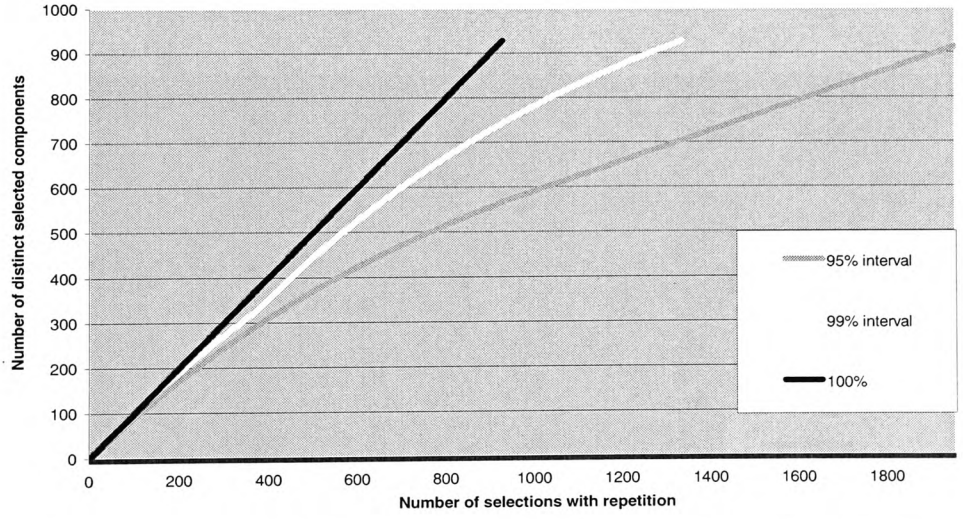

Figure 10.3: *The* 95%, 99% *and* 100% *confidence limits for finding the 928 internal padding components.*

Figure 10.5 shows a comparison between 100% (with or without repetition), 99% and 95% (with repetition) confidence limits for the 1024 Hadamard

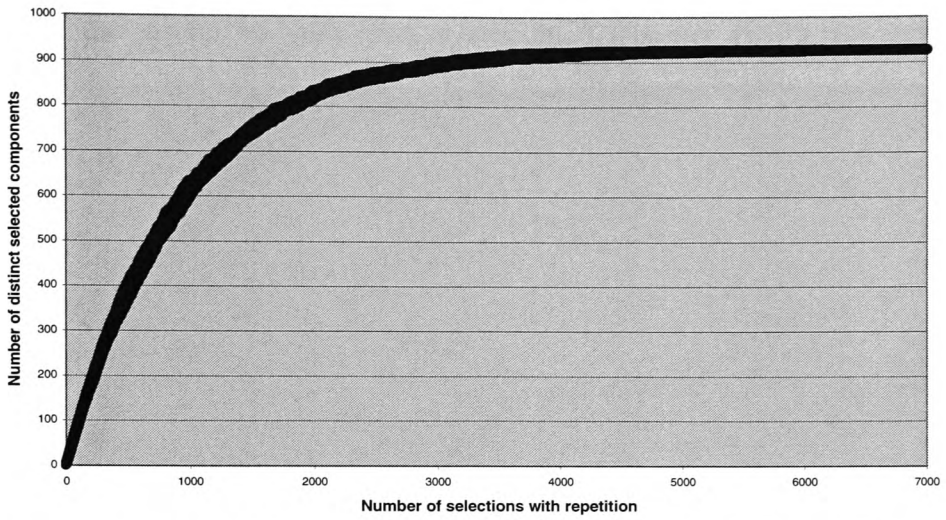

Figure 10.4: *Ten simulations for the number of distinct vectors observed of 928 internal padding vectors. Selection of the next internal padding vector is made with repetition allowed.*

matrix rows.

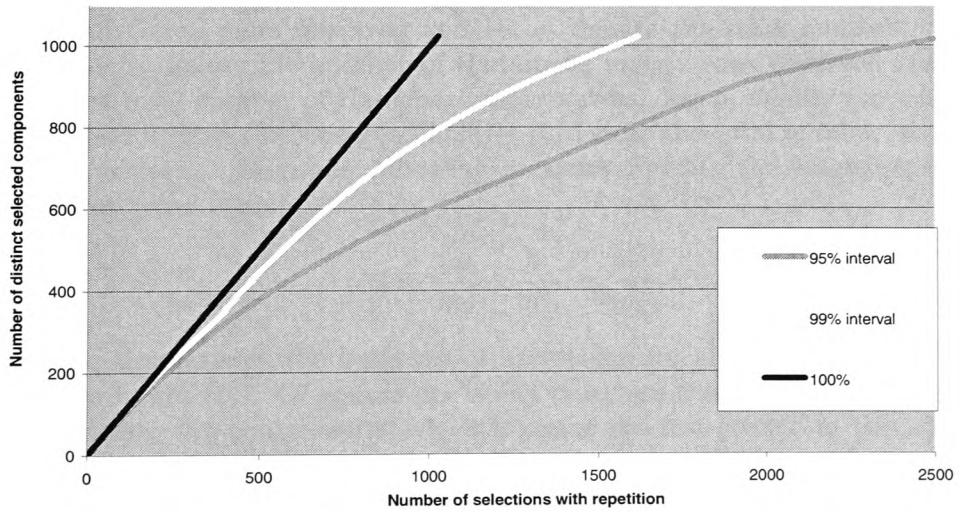

Figure 10.5: *The* 95%, 99% *and* 100% *confidence limits for finding the 1024 Hadamard matrix rows.*

Figure 10.6 also shows ten simulations of the number of selections to find all 1024 Hadamard matrix rows used in the construction.

It is now useful to work with a combined measure that more accurately

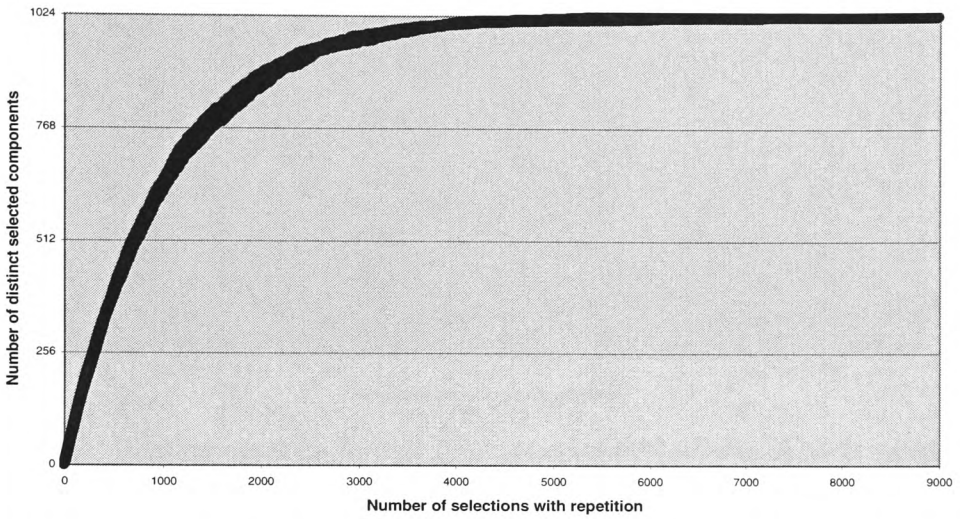

Figure 10.6: *Ten simulations for the number of distinct rows observed of 1024 Hadamard matrix rows. Selection of the next Hadamard matrix row is made with repetition allowed.*

represents the build up of knowledge of all three components. Let g_1 denote the number of Golay pairs observed and let g_2 denote the total number of Golay pairs. Let h_1 denote the number of Hadamard matrix rows observed and let h_2 denote the total number of Hadamard matrix rows. Let p_1 denote the number of internal padding vectors observed and let p_2 denote the total number of internal padding vectors. Then the following measure weights the knowledge of the component types equally:

$$M = \frac{g_1}{3g_2} + \frac{h_1}{3h_2} + \frac{p_1}{3p_2}$$

Using ten simulations, the build up of knowledge for all three components can be seen in figure 10.7. Of course the Golay pairs are learned much more quickly then the other two components, which explains the fast growth at the start.

10.3 Correlation of observed and predicted codewords

In this section simulations are described that implement one possible type of prediction of the current codeword from the part of the codeword observed so

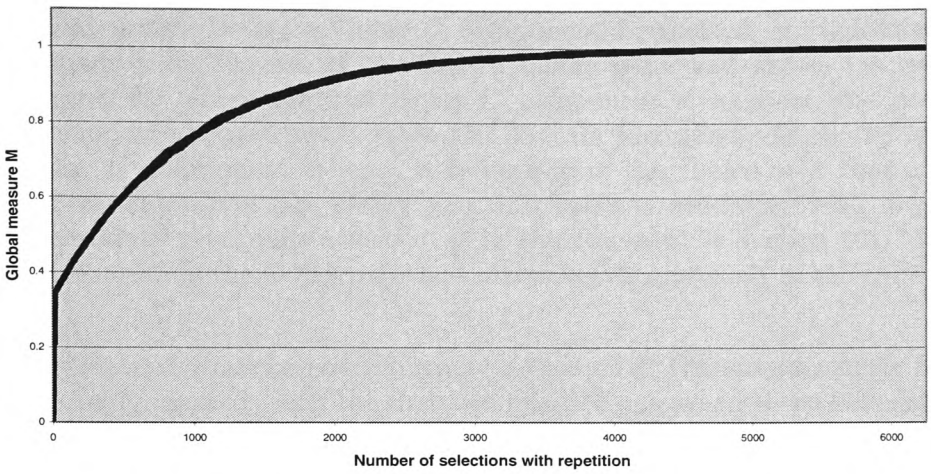

Figure 10.7: *Ten simulations for all three components combined with repetition allowed.*

far. Graphs will be presented showing how the correlation between the predicted codeword and the true codeword changes as the true codeword is read. This will be done both with the assumption of no knowledge of the components of the LS construction in use during the evolution and with the assumption of full knowledge.

10.3.1 Description of the "no knowledge of the components in use" simulation

1. **Internal padding:** Initially an assumed vector of internal padding lengths is constructed randomly from the available lengths. Once the first actual internal padding length is observed, then if it is not equal to the first assumed length, an interchange is performed with a later occurrence of the observed length. For example:

   First randomly assign the padding lengths

   2 0 4 6 4 ...

   If the length 6 is observed first then an interchange is performed:

   6 0 4 2 4 ...

   This continues with successive comparisons of components of the internal padding vectors until all the true padding lengths have been observed, or the internal padding vector can be uniquely completed.

2. **Golay pair:** Before a Golay C component is received, a random choice is made from the set of 384 known Golay pairs and mates (as seen in chapter 4). Once the first Golay C component is received, the possible S components and mates from the 384 are identified. Once the second Golay C component is read, it follows from the choice of π that all the information about the Golay pair and mate is available (with any sign uncertainty eventually resolved as in the comment in section 9.3). Thus, information of the Golay pairs and mates is achieved early in the codeword.

3. **Hadamard matrix row:** Initially an assumed Hadamard matrix row is randomly created, with the first two positions assumed to be +1. As each Golay C component is read, the true sign of the corresponding row position is identified. This progressively predicts the true Hadamard matrix row. For example:

   Initially randomly selected row:

   $$+ \; + \; + \; - \; - \; + \; \ldots$$

   As the Golay C components $c_0, c_1, -c_1, c_0, \ldots$ are received:

   $$\begin{aligned} c_0 \text{ received} &: + \; + \; + \; - \; - \; + \; \ldots \\ c_1 \text{ received} &: + \; + \; + \; - \; - \; + \; \ldots \\ -c_1 \text{ received} &: + \; + \; - \; - \; - \; + \; \ldots \\ c_0 \text{ received} &: + \; + \; - \; + \; - \; + \; \ldots \end{aligned}$$

Simulation 1

Read the vector π
Initialise a choice of internal padding vector of length $p-1$ (IP)
Initialise a choice of a Golay pair and its mate of length N $(\mathbf{c}_0, \mathbf{c}_1, \mathbf{s}_0, \mathbf{s}_1)$
Initialise a choice of Hadamard matrix row of length p (Had)
Read the true LS code of length n (LS)
Read y *Golay pairs and their mates of length* N $(\mathbf{c}_0^{(i)}, \mathbf{c}_1^{(i)}, \mathbf{s}_0^{(i)}, \mathbf{s}_1^{(i)}$ i $\in \{1, 2, \ldots, y\})$
Use initial choices for components to create LSguess;
Find correlation between LSguess *and* LS;
currentposition = *start of the first* C *component;*
For j *from* 0 *to* $p-1$ *do*
    currentIP = 0;
    while LS[currentposition] = 0 *do*
        currentIP = currentIP + 1;
        currentposition = currentposition + 1;

```
end while
if j ≥ 1 and IP[j] ≠ currentIP then
    for k from j + 1 to p − 2 do
        if IP[k] = currentIP then
            IP[k] = IP[j];
            IP[j] = currentIP;
            exit for loop
        end if
    end for
end if
if j = 0 then
    for m from 1 to N do
        c0[m] = LS[currentposition];
        currentposition = currentposition + 1;
    end for
    for t from 1 to y do
        if c0^(t) = c0 then
            store position t of c0 in list of Golay pairs;
        end if
    end for
    Had[0] = +1;
end if
if j = 1 then
    for m from 1 to N do
        c1[m] = LS[currentposition];
        currentposition = currentposition + 1;
    end for
    for r from 1 to y do
        if c1^(r) = c1 then
            store position r of c1 in list of Golay pairs;
        end if
    end for
    Had[1] = +1;
end if
if j > 1 then
    for m from 1 to N do
        currentC[m] = LS[currentposition];
        currentposition = currentposition + 1;
    end for
    if π[j] = 0 then
        if currentC = c0 then
            Had[j] = +1;
```

```
            else
                Had[j] = −1;
            end if
        else
            if currentC = c1 then
                Had[j] = +1;
            else
                Had[j] = −1;
            end if
        end if
    end if
    Use the choices for components to create LSguess;
/* Sign of the second half of the code assumed correct */
    Find correlation between LSguess and LS;
end for
/* No need to examine the second half of the code */
```

Note that it is possible to generate the second half of the code with the opposite sign throughout. If (C, S) is a Golay pair then $(C, -S)$ is also a Golay pair. However if we assume that the third party could attempt jamming with both of these cases simultaneously, it is reasonable to assume that the correct sign is chosen.

10.3.2 Description of the "full knowledge of the components in use" simulation

In this simulation the observer is assumed to know all of the components in use during evolution. However, the actual component of each type currently being used is not known initially.

1. **Internal padding:** Initially an assumed vector of internal padding lengths is randomly selected from the list of vectors known to be in use. Once the first internal padding length is observed, it is possible to reduce the available set of vectors by matching the first position, and then a random selection is made from the reduced list. Once the second actual padding length is observed, the available set of vectors is further reduced and a new random selection is made. This process is continued until only one internal padding vector remains. For example,

   First randomly select the padding vector

   2 0 4 6 4 ...

When the length 6 is observed first select the possible vectors:

6 7 0 1 0 ...

6 2 4 3 2 ...

6 1 0 0 1 ...

6 6 3 8 6 ...

One is selected at random:

6 2 4 3 2 ...

When the length 1 is observed second, the list is further reduced:

6 1 0 0 1 ...

Here this vector must be selected.

2. **Golay pair:** This is identical to the "no knowledge" simulation except that the choice of the first C components is made from a much smaller set.

3. **Hadamard matrix row:** Initially an assumed Hadamard matrix row is selected from the set of Hadamard matrix rows in use. As each Golay C component is read, the available set of rows to select from is reduced. This process is continued until only one Hadamard matrix row remains.

Simulation 2

Read the vector π
Read the true LS code of length n (LS)
Read x *internal padding vectors of length* $p-1$ ($\mathrm{IP}^{(\mathrm{i})}$ $\mathrm{i} \in \{1, 2, \ldots, \mathrm{x}\}$)
Read y *Golay pairs and their mates of length* N ($\mathbf{c}_0^{(\mathrm{j})}, \mathbf{c}_1^{(\mathrm{j})}, \mathbf{s}_0^{(\mathrm{j})}, \mathbf{s}_1^{(\mathrm{j})}$ $\mathrm{j} \in \{1, 2, \ldots, \mathrm{y}\}$)
Read z *Hadamard matrix rows of length* p ($\mathrm{Had}^{(\mathrm{k})}$ $\mathrm{k} \in \{1, 2, \ldots, \mathrm{z}\}$)
Generate a random number $a \in \{1, 2, \ldots, x\}$
Initialise a choice of internal padding ($\mathrm{IP}^{(\mathrm{a})}$)
Generate a random number $b \in \{1, 2, \ldots, y\}$
Initialise a choice of a Golay pair and its mate of length N ($\mathbf{c}_0^{(\mathrm{b})}, \mathbf{c}_1^{(\mathrm{b})}, \mathbf{s}_0^{(\mathrm{b})}, \mathbf{s}_1^{(\mathrm{b})}$)
Generate a random number $c \in \{1, 2, \ldots, z\}$
Initialise a choice of Hadamard matrix row ($\mathrm{Had}^{(\mathrm{c})}$)
Use initial choices for components to create LSguess*;*
Find correlation between LSguess *and* LS*;*
currentposition = *start of first* C *component;*
IPchoicearray = *array of all* x *internal padding vectors;*
Hadchoicearray = *array of all* z *Hadamard matrix rows;*
for j *from* 0 *to* $p-1$ *do*

currentIP = 0;
while LS[currentposition] = 0 *do*
    currentIP = currentIP + 1;
    currentposition = currentposition + 1;
end while
if $j \neq 0$ *then*
    for k *from* 1 *to* x *do*
        if $\mathrm{IP}^{(k)}[j] \neq$ currentIP *then*
            delete $\mathrm{IP}^{(k)}$ *from* IPchoicearray;
        end if
    end for
end if
select (IP) *at random from* IPchoicearray;
if $j = 0$ *then*
    for m *from* 1 *to* N *do*
        currentC[m] = LS[currentposition];
        currentposition = currentposition + 1;
    end for
    for t *from* 1 *to* y *do*
        if $\mathbf{c}_0^{(t)} = \mathbf{c}_0$ *then*
            $\mathbf{c}_0$, $\mathbf{c}_1^{(t)}$, $\mathbf{s}_0^{(t)}$, $\mathbf{s}_1^{(t)}$ *are the Golay pair components in use;*
        end if
    end for
    $Had[0] = +1$;
end if
if $j > 0$ *then*
    for m *from* 1 *to* N *do*
        currentC[m] = LS[currentposition];
        currentposition = currentposition + 1;
    end for
    if $\pi[j] = 0$ *then*
        if currentC = $\mathbf{c}_0$ *then*
            $currentHad = +1$;
        else
            $currentHad = -1$;
        end if
    else
        if $currentC = \mathbf{c}_1$ *then*
            $currentHad = +1$;
        else
            $currentHad = -1$;
        end if

```
        end if
        for e from 1 to z do
            if Had^(e)[j] ≠ currentHad then
                delete Had^(e) from Hadchoicearray;
            end if
        end for
        select (Had) at random from Hadchoicearray;
    end if
    Use the choices for components to create LSguess;
/* Sign of the second half of the code assumed correct */
    Find correlation between LSguess and LS;
end for
/* No need to examine the second half of the code */
```

10.3.3 Presentation of graphs

Here graphs will be presented showing the results of 100 simulations with both the assumption of no knowledge of the components of the LS construction in use during the evolution and with the assumption of full knowledge, as described in section 10.3.1 and 10.3.2. In figures 10.8 and 10.10 it will be shown how the average correlation (over the 100 runs) between the predicted codeword and the true codeword changes as the true codeword is read. Figures 10.9 and 10.11 show distributions of the number of chips before 25%, 50%, 75% and 100% correlation is achieved.

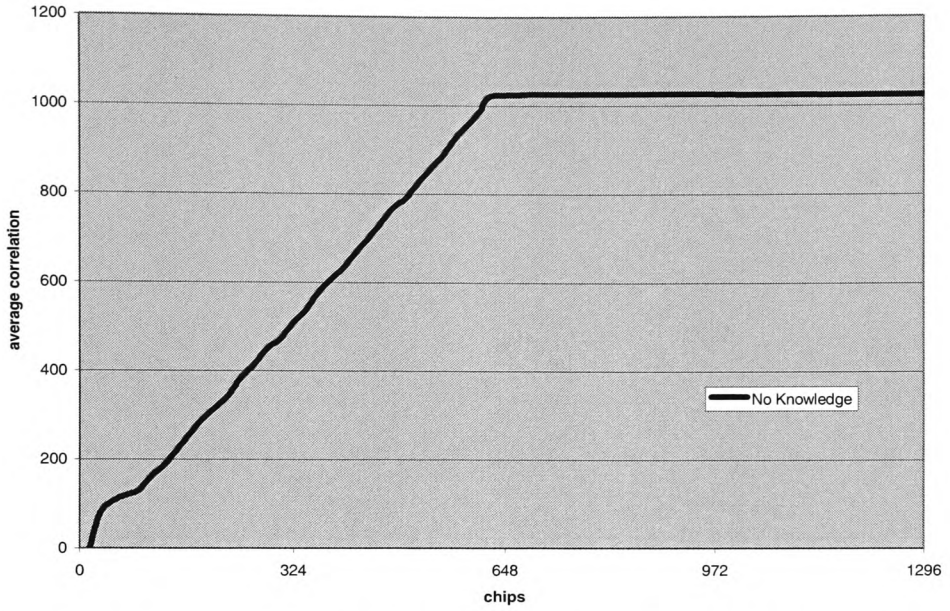

Figure 10.8: *The average correlation over 100 simulations for the "no knowledge" case.*

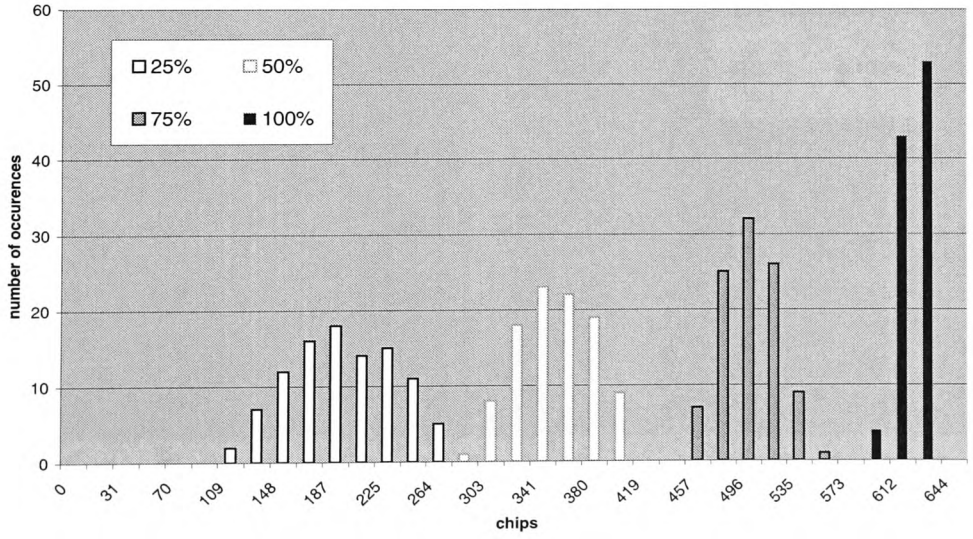

Figure 10.9: *Distributions of correlations over the 100 simulations for the "no knowledge" case.*

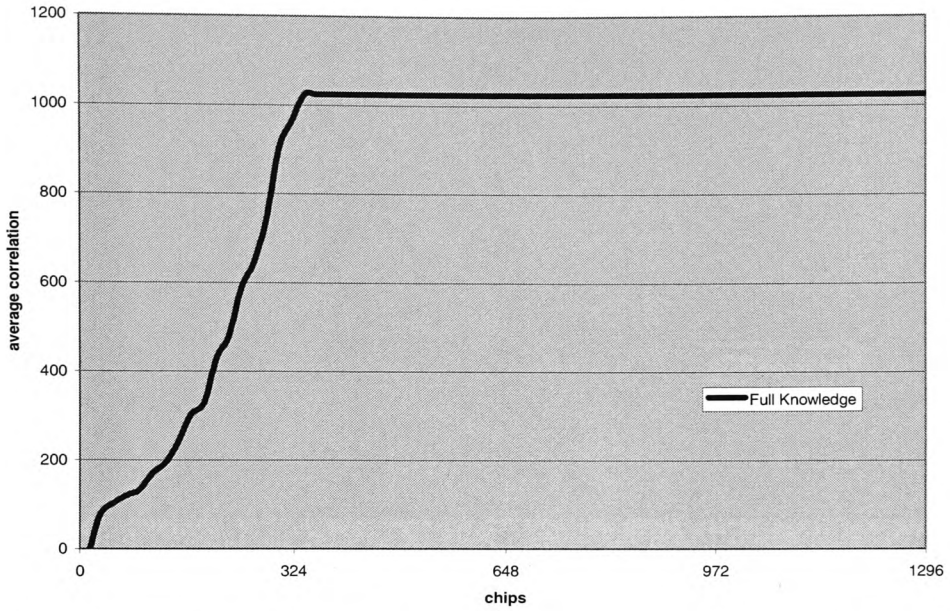

Figure 10.10: *The average correlation over 100 simulations for the "full knowledge" case.*

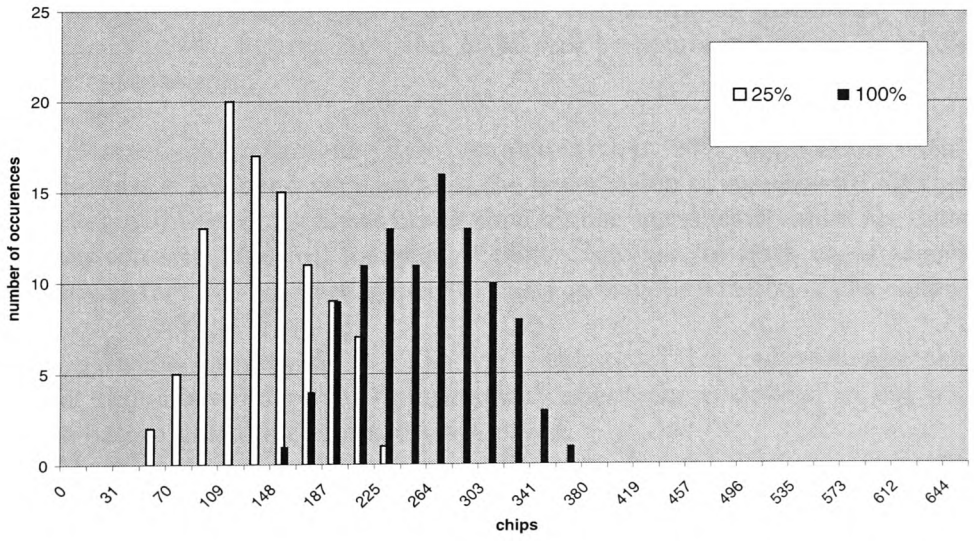

Figure 10.11: *Distributions of correlations over the 100 simulations for the "full knowledge" case. In most runs the correlation jumps from about* 25% *to* 100% *without intermediate values being observed.*

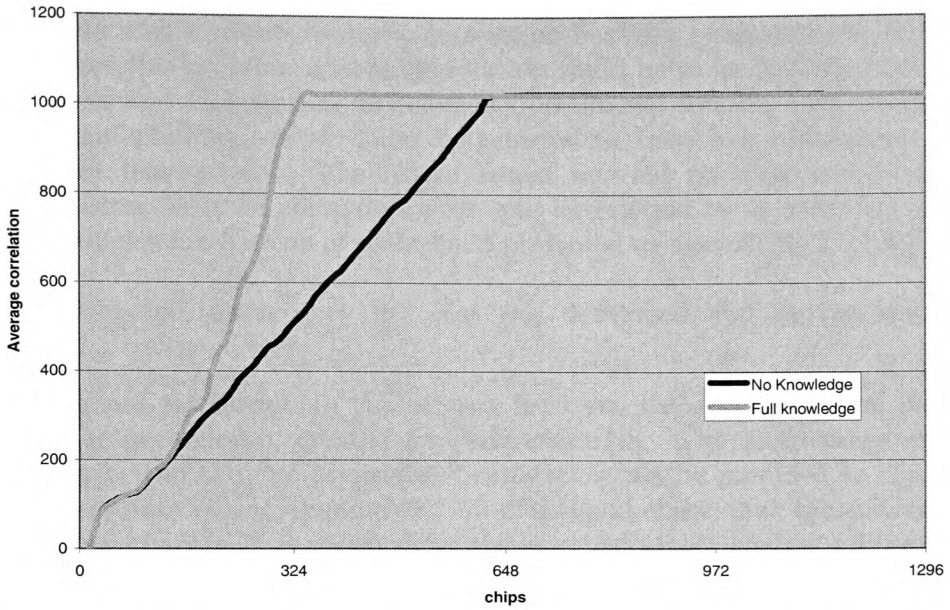

Figure 10.12: *The average correlations over 100 simulations for the "no knowledge" and "full knowledge" cases.*

10.3.4 Comparison of graphs

In this section the graphs shown in section 10.3.3 will be discussed. For ease of reference the two figures 10.8 and 10.10 will be combined into a single figure 10.12 for comparison.

The clearest indication of these graphs is that 25% correlation with the true codeword is achieved very early in the observation of a codeword. If there is no knowledge of the components in use then higher correlation values are deferred in comparison with the full knowledge case. However, in both cases even high correlation values are achieved relatively early in the observation of the codeword.

This is perhaps unsurprising. The very nature of LS codes means that it is certain that essentially "full information" about the codeword in use will be available before half the codeword is observed.

10.4 A recency based approach to the generation of internal padding lengths

Ideally the code should evolve in such a way that no knowledge can be gained of the components used. This conflicts with the need to define specific sets of

components which ensure that the correlation between generations is bounded. To overcome this problem, a possible solution could be to let just the Hadamard matrix rows and Golay pairs be selected from known sets for each generation. The internal padding vector could be selected to have low correlation values with recent generations. The list of recent internal padding vectors that a possible vector must be compared with, will be referred to as a *recency list.* A detailed recency mechanism of this type is presented as mechanism 2 in chapter 8.

A simulation for the recency list case was developed and can be described as follows:

1. **Internal padding:** In the recency list case, the actual internal padding vector is randomly created for each evolution. The interchange method described in the "no knowledge" simulation can be modified to check the appropriate recency condition. An additional check that the criterion for χ_3 (see chapter 7) is satisfied for the assumed vector against all vectors in the recency list is included. If the criterion is not satisfied for any vector in the list, a further permutation of the remaining assumed internal padding vector entries is made until it is satisfied for all vectors.

2. **Golay pair:** This is identical to the "full knowledge" simulation.

3. **Hadamard matrix row:** This is identical to the "full knowledge" simulation.

Simulation 3

Read the vector π
Read the true LS code of length n (LS)
Read previous x *internal padding vectors of length* $p-1$ ($\mathrm{IP}^{(\mathrm{i})}$ $\mathrm{i} \in \{1, 2, \ldots, \mathrm{x}\}$)
Initialise the maximum number of coincidences (MaxCo) *between the internal padding vectors*
Read y *Golay pairs and their mates of length* N ($\mathbf{c}_0^{(\mathrm{j})}, \mathbf{c}_1^{(\mathrm{j})}, \mathbf{s}_0^{(\mathrm{j})}, \mathbf{s}_1^{(\mathrm{j})}$ $\mathrm{j} \in \{1, 2, \ldots, \mathrm{y}\}$)
Read z *Hadamard matrix rows of length* p ($\mathrm{Had}^{(\mathrm{k})}$ $\mathrm{k} \in \{1, 2, \ldots, \mathrm{z}\}$)
Initialise a choice of internal padding (IP) *such that the correlation between the new* IP *and all previous* x *in the recency list is below* $\chi_3 = MaxCo/p$
Generate a random number $b \in \{1, 2, \ldots, y\}$
Initialise a choice of a Golay pair and its mate of length N ($\mathbf{c}_0^{(\mathrm{b})}, \mathbf{c}_1^{(\mathrm{b})}, \mathbf{s}_0^{(\mathrm{b})}, \mathbf{s}_1^{(\mathrm{b})}$)
Generate a random number $c \in \{1, 2, \ldots, z\}$
Initialise a choice of Hadamard matrix row ($\mathrm{Had}^{(\mathrm{c})}$)
Use initial choices for components to create LSguess*;*
Find correlation between LSguess *and* LS*;*
currentposition = *start of first* C *component;*
Hadchoicearray = *array of all* z *Hadamard matrix rows;*

for j *from* 0 *to* $p-1$ *do*
    currentIP = 0;
    while LS[currentposition] = 0 *do*
        currentIP = currentIP + 1;
        currentposition = currentposition + 1;
    end while
    if $j > 0$ *then*
        IPok = FALSE;
        IP[j] = currentIP;
        while IPok = FALSE *do*
            for k *from* $j+1$ *to* $p-2$ *do*
                randomly assign remaining choices to create vector IP;
            end for
            coincidence = 0;
            for i *from 1 to* x *do*
                coincidence = max{coincidence, maximum possible number of coincidences between IP and $IP^{(i)}$};
            end for
            if coincidence $\leq$ MaxCo *then*
                IPok = TRUE;
            end if
        end while
    end if
    if $j = 0$ *then*
        for m *from* 1 *to* N *do*
            currentC[m] = LS[currentposition];
            currentposition = currentposition + 1;
        end for
        for t *from* 1 *to* y *do*
            if $\mathbf{c}_0^{(t)} = \mathbf{c}_0$ *then*
                $\mathbf{c}_0$, $\mathbf{c}_1^{(t)}$, $\mathbf{s}_0^{(t)}$, $\mathbf{s}_1^{(t)}$ *are the Golay pair components in use;*
            end if
        end for
        $Had[0] = +1$;
    end if
    if $j > 0$ *then*
        for m *from* 1 *to* N *do*
            currentC[m] = LS[currentposition];
            currentposition = currentposition + 1;
        end for
        if $\pi[j] = 0$ *then*
            if currentC = $\mathbf{c}_0$ *then*

```
                currentHad = +1;
            else
                currentHad = −1;
            end if
        else
            if currentC = c1 then
                currentHad = +1;
            else
                currentHad = −1;
            end if
        end if
        for e from 1 to z do
            if Had(e)[j] ≠ currentHad then
                delete Had(e) from Hadchoicearray;
            end if
        end for
        select (Had) at random from Hadchoicearray;
    end if
    Use the choices for components to create LSguess;
/* Sign of the second half of the code assumed correct */
    Find correlation between LSguess and LS;
end for
/* No need to examine the second half of the code */
```

Figure 10.13 shows (for $p = 32$ and $N = 16$) how the average correlation (over the 100 runs, with a recency list of length 50) between the predicted codeword and the true codeword changes as the true codeword is read using the recency approach. Figure 10.14 shows distributions of the number of chips before 25%, 50%, 75% and 100% correlation is achieved for the recency approach. Figure 10.12 and 10.13 are combined in figure 10.15 to allow all three cases to be compared. These graphs show that with only the set of internal padding vectors unknown to the observer, correlation values are similar to the "no knowledge" case.

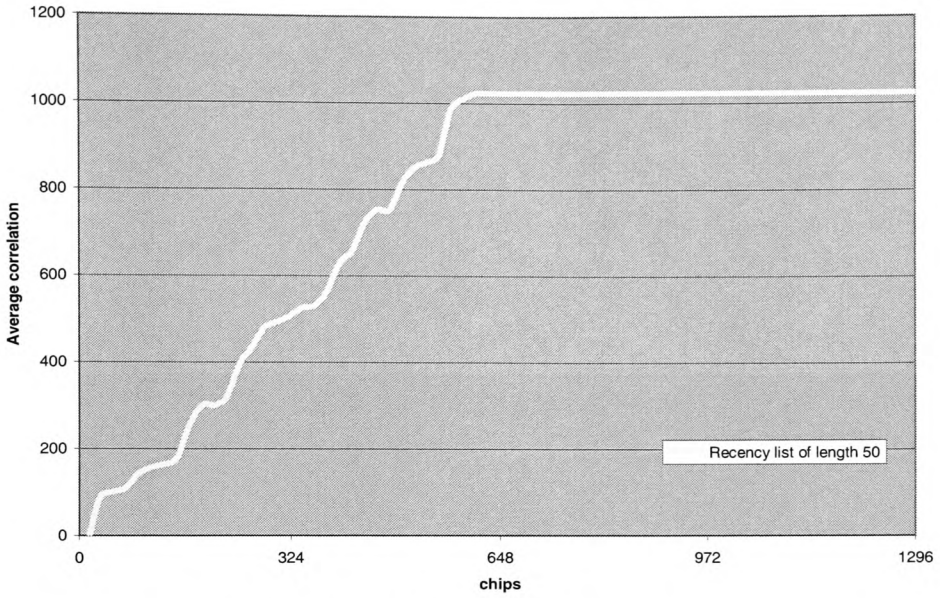

Figure 10.13: *The average correlation over 100 simulations using the recency approach.*

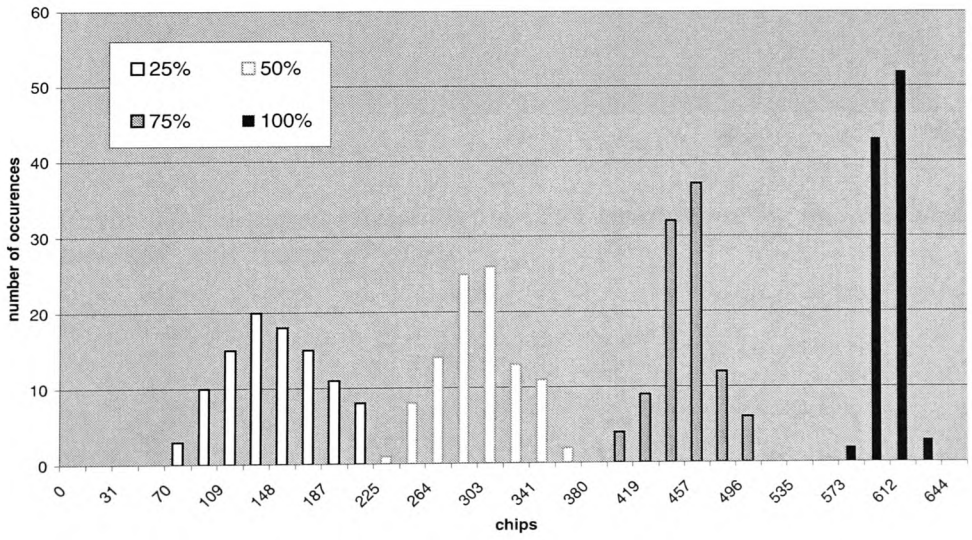

Figure 10.14: *Distributions of correlations over the 100 simulations for the recency approach.*

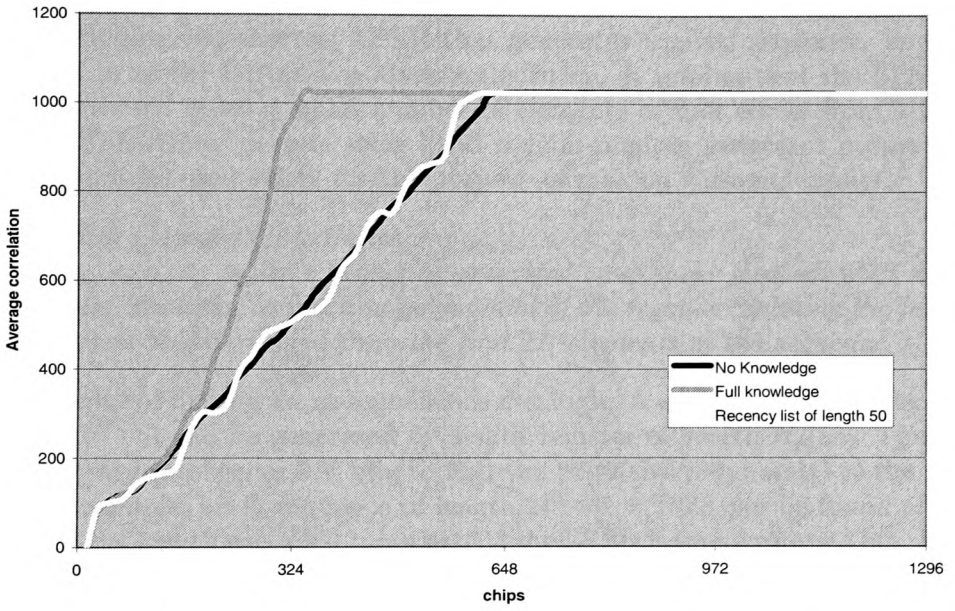

Figure 10.15: *The average correlation over 100 simulations for "no knowledge", "full knowledge" and "recency" cases.*

10.5 Comparison with m-sequences

In this section it will be demonstrated that the evolving LS code is not worse than codes in use today for CDMA. Comparison will be made with m-sequences. An m-sequence is a binary sequence created from a linear feedback shift register (LFSR). An m-sequence creates a pseudo random binary sequence by using a LFSR with a period $2^N - 1$ where N is the size of the LFSR. The binary polynomial defining the shift register must be primitive [14]. This is a simple way of generating an infinite binary string that has a non trivial structure. It is commonly used as a pseudo random number generator for its simplicity and because it has some desirable properties for a random sequence. Berlekamp discovered in 1968 an algorithm [1] for decoding BCH codes. Then in 1969 it was interpreted by Massey [15] as a way of finding the shortest LFSR that generates a given sequence, and so it is referred to as the Berlekamp-Massey algorithm. It implies that the LFSR can be reconstructed given a small number of elements of first terms from a binary sequence. Therefore, despite some good pseudo-random generator properties, a LFSR cannot be used safely for the purpose of making a stream cipher.

Theorem 8 *(Massey [15], Berlekamp [1])*
Let $s = s_0, s_1, \ldots, s_n$ be a sequence generated by a linear feedback shift register with L cells. Then the connection polynomial of the register (defining the feedback positions) can be determined from the first $2L$ elements of the sequence.

The difficulty of finding an m-sequence is not high. A maximum length sequence of length $2^N - 1$ can be generated by a shift register of length N [18]. Then it is only necessary to observe $2N$ bits to find the primitive polynomial of the LFSR [15]. For example, an m-sequence of length $2^{10} - 1 = 1023$ can be found after 20 bits and if the length was $2^{22} - 1 = 4194303$ the LFSR is found after 44 bit. Figure 10.16 displays Maple style pseudo-code for the Berlekamp-Massey algorithm. An example of the Maple output from this algorithm is given in figure 10.17. Figure 10.18 demonstrates that evolving LS codes compare favourably with m-sequences.

10.6 Comparison with Gold sequences

In practice Gold sequences are often used in CDMA applications, as the number of sequences with reasonably satisfactory correlation properties is large. A Gold sequence of length $2^N - 1$ can be generated by a shift register of length $2N$ [18]. Thus it is only necessary to observe $4N$ bits to find the polynomial of the LFSR [15]. Figure 10.19 demonstrates that evolving LS codes compare favourably with Gold codes.

```
Read the sequence s generated by a LFSR with N cells
Initialise B = 1, C = 1, L = 0, k = 1, b = 1;
for n from 0 to 2N + 6 do        /*+6 to show convergence */
    d = s[n + 1];
        for i from 1 to L do
            d = d + coeff(C, x^i) * s[n − i + 1];
        od;
    if d = 0 then
        k = k + 1;
    fi;
    if (d <> 0 and 2 * L > n) then
        C = expand(C − d * x^k * B/b);
        k = k + 1;
    fi;
    if (d <> 0 and 2 * L <= n) then
        T = C;
        C = expand(C − d * x^k * B/b);
        B = T;
        L = n + 1 − L;
        k = 1;
        b = d;
    fi;
    Print C;
od;
```

Figure 10.16: *Maple style pseudo-code for the Berlekamp-Massey algorithm.*

```
Input: s={0 0 1 1 0 1 0 0 1 0 0 0 0 1 0 1 0 1 1 1 0 1 1 0 0 0 1 1 1 1 1}:
Output:

1
1
1+x^3
1+x+x^3
1+x+x^2+x^3
1+x+x^2
1+x+x^2+x^4
1+x+x^2+x^4
1+x+x^3
1+x^2+x^5
1+x^2+x^5
1+x^2+x^5
1+x^2+x^5
1+x^2+x^5
1+x^2+x^5
```

Figure 10.17: *Output of the Berlekamp-Massey algorithm in figure 10.16.*

10.7 Conclusions

Figure 10.15 confirms that complete security against jamming is unachievable. Consider a threshold of 25% correlation, as assumed earlier in the thesis. It can be seen that, even with no knowledge of the sets of components in use during evolution, this threshold is achieved very early in the reading of a single codeword. However the challenge to the third party of actually performing the necessary computations on this time scale is daunting.
Define two time intervals:

1. The third party latency $TL(\tau, \text{cor})$ is the time taken by the third party to predict the true codeword given that the codeword is known up to and including chip τ, and that the predicted codeword and the true codeword must have a correlation of at least cor.

2. The system latency SL is the time taken by a user to create a new codeword after evolution.

The security of the system in practice depends on the value of the third party latency. As long as the third party latency $TL(0, \text{cor})$ exceeds the length of time necessary to transmit a single bit, the system will be secure provided the system latency is such that the code can evolve after every bit is transmitted. Of course it may be possible to perform the computations

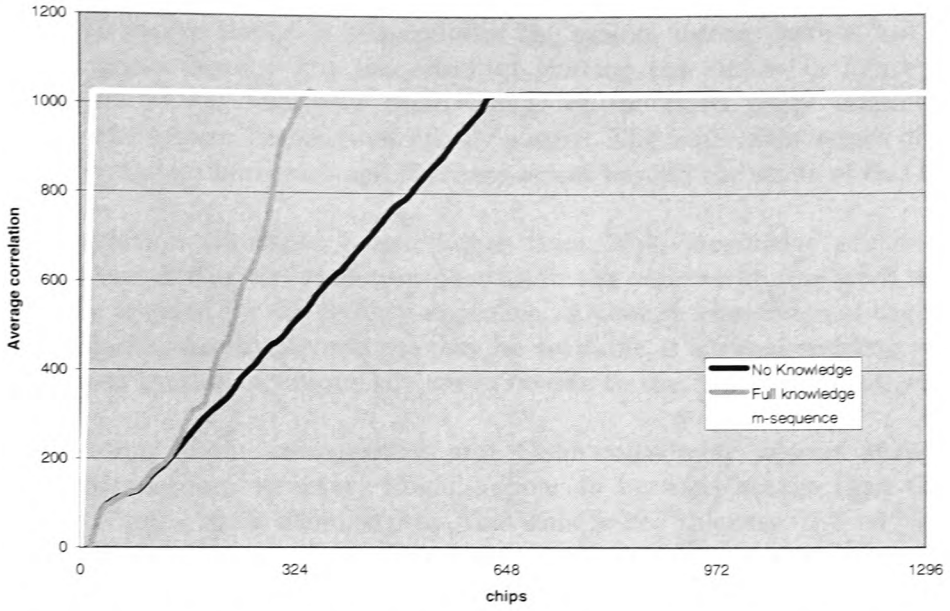

Figure 10.18: *A comparison of the "no knowledge" and "full knowledge" cases of an LS code with an m-sequence.*

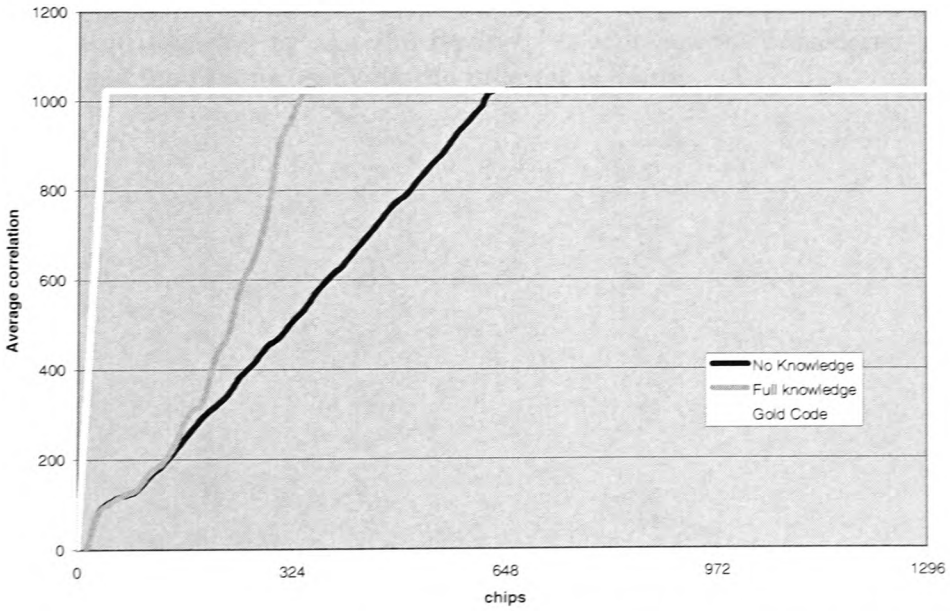

Figure 10.19: *A comparison of the "no knowledge" and "full knowledge" cases of an LS code with a Gold code.*

necessary to evolve the code in parallel if the system latency proves too large. The third party latency has the effect of shifting the curves in figure 10.15 to the right. Even relatively small values of the third party latency may mean that the system becomes effectively secure. The achievable values of these latencies depend on hardware and firmware issues beyond the scope of this thesis.

For a correlation threshold much higher than 25%, knowledge of the components in use during the evolution does allow the code to be predicted earlier. This can be avoided by the recency approach. Although knowledge of the Golay pairs and Hadamard matrices in use may be available, if internal padding vectors are generated by this technique the curve reverts to the "no knowledge" curve.

The conclusion about m-sequences and Gold codes may appear surprising. Their pseudo-random structure might appear to be more secure than that of LS codes. Figure 10.18 demonstrates that this is not the case. LS codes have greater potential for security provided the third party latency is not too small. This arises because LS codewords are not generated by a linear feedback shift register and so the Berlekamp Massey algorithm does not apply. On the other hand not using a LFSR may very significantly increase the system latency.

The recency based approach has been shown to be successful and is not susceptible to learning by the third party. It will now be considered as the recommended method for evolving the internal padding.

Chapter 11

Further examples

Most of the work in this thesis has been illustrated by a single standard example. In this chapter the range of examples will be expanded. Table 11.1 is based on that in [17] and shows a variety of LS codes that might be useful in practical application.

The two parameters to be varied are p and N. The parameter p re-

N	p	No. of codewords	(Tr,NTr)	No. of ± 1's	Maximum repetitions	Total internal padding	External padding	Length n	Duty ratio	Normalized correlation
8	16	32	(8,7)	256	2	98	30	384	0.67	0.0625
8	16	32	(15,0)	256	2	98	30	384	0.67	0.125
8	32	64	(16,15)	512	4	210	30	752	0.68	0.0625
8	32	64	(24,7)	512	4	210	30	752	0.68	0.09375
8	32	64	(31,0)	512	4	210	30	752	0.68	0.125
8	64	128	(32,31)	1024	8	434	30	1488	0.69	0.0625
8	64	128	(48,15)	1024	8	434	30	1488	0.69	0.09375
8	64	128	(63,0)	1024	8	434	30	1488	0.69	0.125
16	16	32	(8,7)	512	2	98	62	672	0.76	0.0625
16	16	32	(12,3)	512	2	98	62	672	0.76	0.09375
16	16	32	(15,0)	512	2	98	62	672	0.76	0.125
16	32	64	(16,15)	1024	4	210	62	1296	0.79	0.0625
16	32	64	(24,7)	1024	4	210	62	1296	0.79	0.09375
16	32	64	(31,0)	1024	4	210	62	1296	0.79	0.125
16	64	128	(32,31)	2048	8	434	62	2544	0.81	0.0625
16	64	128	(48,15)	2048	8	434	62	2544	0.81	0.09375
16	64	128	(63,0)	2048	8	434	62	2544	0.81	0.125
32	16	32	(8,7)	1024	2	98	126	1248	0.82	0.0625
32	32	64	(16,15)	2048	4	210	126	2384	0.86	0.125

Table 11.1: *Duty ratio (number of ± 1's divided by length) and normalized correlations for various parameters. "Maximum repetitions" is the maximum number of repetitions in $\{L_1, L_2, L_3, \ldots, L_{p-1}\}$. "Normalized correlation" is the maximum aperiodic correlation with $|\tau| \leq 2N - 1$ divided by the number of ± 1's in a codeword.*

quires study from the point of view of the Hadamard matrices and the internal

padding vectors. The parameter N requires study from the point of view of the Golay pairs. The value $p = 32$ has already been studied. Here the values $p = 16$ and $p = 64$ will be studied from both points of view. Similarly, the value $N = 16$ (and $N = 8$ for single overlap computations only) have been studied. Here the values $N = 8$ (double overlap computation) and $N = 32$ will be studied.

11.1 The sets of Hadamard matrices

The results of chapter 5 with Gold or Gold-like constructions are important if it is to be difficult to predict the Hadamard matrix in use from knowledge of one Hadamard matrix row in use. This is necessary if the system is to be secure against the loss of a radio (see section 5.3). For both $p = 32$ and $p = 64$ the construction leads to a peak normalized correlation threshold of 0.25. For $p = 16$ the peak normalized correlation threshold is actually 0.5 although this peak is achieved by relatively few pairs of codewords and so can be regarded as acceptable, with a threshold of 0.25 still used for the other components.

11.2 The sets of Golay pairs

Here the double overlap computations of chapter 7 are repeated for $N = 8$ and $N = 32$.

For $N = 8$, a clique of size 1 is obtained for $\chi_2^{(3)} = 0.375$ and a clique of size 13 is obtained for $\chi_2^{(3)} = 0.5$ (shown in table 11.2). Note that if one of these cliques is used, the correlation bound applies to the two generations of LS code, whatever the Hadamard matrix, internal padding vector or π vector (even if the Hadamard matrix row and internal padding vector are the same in the two generations).

For $N = 32$, a clique of size 1 is obtained for $\chi_2^{(3)} = 0.25$, 0.3125 and 0.375 (see table 11.2).

The general conclusion is that changes of Golay pair without changing the Hadamard matrix row or internal padding vector are not sufficient to ensure normalized correlations of at most 0.25. It still may be useful to use a larger set of Golay pairs, provided the internal padding vector is forced to change between close generations.

N	σ	*Size of clique*
8	1	192
8	0.5	13
8	0.375	1
8	0.25	1
16	1	384
16	0.375	5
16	0.3125	2
16	0.25	1
32	1	7680
32	0.375	1
32	0.3125	1
32	0.25	1

Table 11.2: *The clique size for various thresholds σ of peak normalized aperiodic cross-correlation using the double overlap clique search for Golay pairs. N is the length of the Golay pair.*

11.3 The sets of internal padding vectors

The use of the recency based construction has been shown to be attractive. Here the computations leading to tables 8.1 and 8.2 of chapter 8 will be repeated for $p = 16$ and $p = 64$ and presented in tables 11.3, 11.4, 11.5 and 11.6.

Table 11.3 shows for $p = 16$ the percentage of accepted randomly generated internal padding vectors for single trials ($\beta = 1$) with various recency list sizes. Table 11.4 shows for $p = 16$ the percentage of trials rejected when $\beta > 1$. Here a single trial is a maximum of β iterations. The mean and maximum normalized correlation for the rejected cases is also shown.

With a recency list of length 1, almost all new internal padding vectors are accepted (95.1%), whereas with a recency list of length 50, only 23.9% are accepted. Although the results are a little worse than for $p = 32$, it can be seen from table 11.4 that a recency list of size up to 50, with β between 5 and 10 still seems satisfactory for $p = 16$. The number of outliers with a normalized correlation approaching 0.5 is in fact very small.

Now consider $p = 64$. Table 11.5 shows the percentage of accepted randomly generated internal padding vectors for single trials ($\beta = 1$) with various recency list sizes. Table 11.6 shows the percentage of trials rejected when $\beta > 1$. Here a single trial is a maximum of β iterations. The mean and maximum

Recency list size	*No. accepted*	*No. rejected*	*% Accepted*
1	951229	48771	95.1
10	923076	76924	92.3
15	739441	260559	73.9
20	611790	388210	61.2
30	443877	556123	44.4
40	305216	694784	30.5
50	239287	760713	23.9
60	180336	819664	18.0

Table 11.3: *The number of accepted and rejected internal padding length vectors for $p = 16$ with at most four coincidences (out of 16) with any vector within the recency list. One million vectors are generated in each case.*

normalized correlation for the rejected cases is also shown.

With a recency list of length 1, almost all new internal padding vectors are accepted (96.8%), whereas with a recency list of length 50, only 25.6% are accepted. Although the results are again a little worse than for $p = 32$, it can be seen that a recency list of size up to 50, with β between 5 and 10 still seems satisfactory for $p = 64$. The number of outliers with a normalized correlation approaching 0.5 is in fact very small.

11.4 Simulation of the recency list approach for $p = 16$ and $p = 64$

Using the recency approach described in section 10.4 and illustrated in figure 10.13, it is useful to see how the simulation might change for different values of p. Figure 11.1 shows the results of simulations for the cases $p = 16$, $p = 32$ and $p = 64$ (with $N = 16$) and with a recency list of length 50.

From figure 11.1 it can be seen that the general shape of the curve is qualitatively the same for $p = 16$ or 64 as for the case $p = 32$ which was previously presented in figure 10.13. Thus the same general conclusions hold.

Recency list size	β	*% Rejected after β iterations*	*Mean normalized correlation for those rejected*	*Maximum normalized correlation for those rejected*
10	10	0.178	0.264	0.375
20	10	0.289	0.271	0.375
30	10	0.673	0.288	0.375
40	10	0.881	0.291	0.4375
50	10	0.952	0.298	0.4375
60	10	1.419	0.295	0.4375
10	5	0.315	0.291	0.375
20	5	0.562	0.295	0.375
30	5	0.992	0.301	0.375
40	5	1.231	0.299	0.4375
50	5	1.912	0.301	0.4375
60	5	3.006	0.354	0.4375
10	2	4.776	0.290	0.375
20	2	9.013	0.299	0.4375
30	2	20.213	0.297	0.4375
40	2	26.515	0.305	0.4375
50	2	50.809	0.309	0.5
60	2	63.195	0.350	0.5

Table 11.4: *The percentage of rejected trials after β iterations for $p = 16$. A trial is rejected if each of the β vectors generated has more than four coincidences (out of 16) with some vectors within the recency list. For the rejected cases the mean (best of β trials) and maximum normalized correlation is shown.*

Recency list size	*No. accepted*	*No. rejected*	*% Accepted*
1	967731	32269	96.8
10	919874	80126	92.0
15	820903	179097	82.1
20	631890	368110	63.2
30	463231	536769	46.3
40	320716	679284	32.1
50	256433	743567	25.6
60	166528	833472	16.7

Table 11.5: *The number of accepted and rejected internal padding length vectors for $p = 64$ with at most sixteen coincidences (out of 64) with any vector within the recency list. One million vectors are generated in each case.*

Recency list size	β	*% Rejected after β iterations*	*Mean normalized correlation for those rejected*	*Maximum normalized correlation for those rejected*
10	10	0.162	0.297	0.28125
20	10	0.227	0.298	0.28125
30	10	0.501	0.299	0.3125
40	10	1.058	0.297	0.34375
50	10	1.769	0.300	0.40625
60	10	2.451	0.309	0.34375
10	5	0.021	0.299	0.28125
20	5	0.198	0.298	0.3125
30	5	0.695	0.299	0.34375
40	5	2.101	0.299	0.4375
50	5	2.750	0.307	0.5
60	5	4.982	0.320	0.5
10	2	6.793	0.301	0.4375
20	2	9.861	0.299	0.4375
30	2	14.004	0.302	0.40625
40	2	24.242	0.306	0.46875
50	2	39.749	0.317	0.46875
60	2	46.108	0.311	0.5

Table 11.6: *The percentage of rejected trials after β iterations for $p = 64$. A trial is rejected if each of the β vectors generated has more than sixteen coincidences (out of 64) with some vectors within the recency list. For the rejected cases the mean (best of β trials) and maximum normalized correlation is shown.*

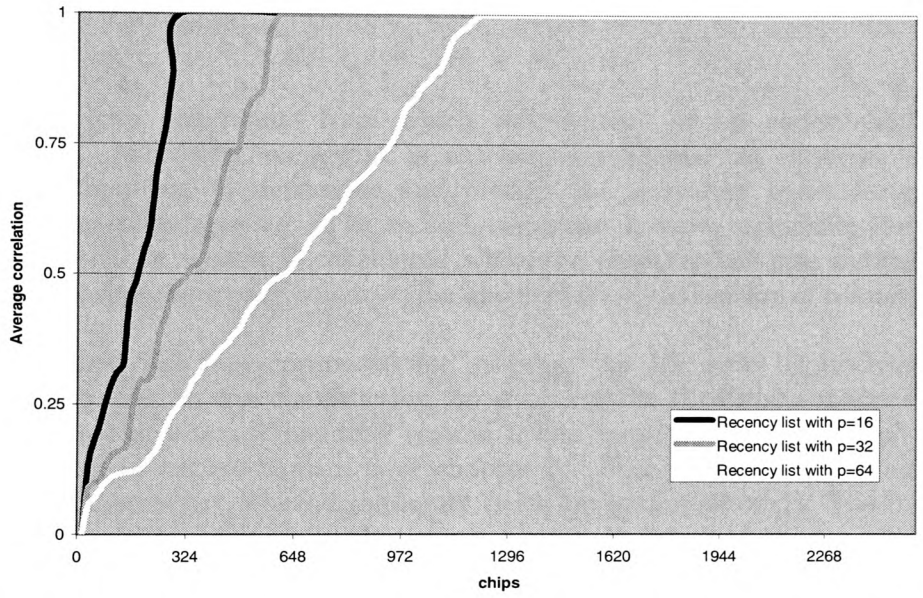

Figure 11.1: *The average normalized aperiodic cross-correlations over 100 simulations when using the recency approach. The recency list has length 50, and results are shown for internal padding vectors of lengths* $p = 16$, $p = 32$ *and* $p = 64$. *In all cases a normalized correlation of 1 is reached just before* $n/2$ *chips.*

Chapter 12

Conclusions

There are two viewpoints from which the security of LS codes should be considered. The first viewpoint is the security against an observer of the system attempting to determine and predict the spreading codes for use in jamming or eavesdropping. The second viewpoint is more stringent; the security of the entire system is considered when the observer has also managed to capture the spreading code construction and evolution mechanism of a single user.

A mechanism has been proposed for evolving the LS code by evolving the Golay pairs used in the construction, by evolving the Hadamard matrices and by evolving the internal padding vectors. The most attractive way of doing this appears to be mechanism 6 of chapter 8. Here the Golay pair and the Hadamard matrix are selected randomly (with repetition allowed). The internal padding vector is selected by the use of a recency list mechanism. A list of possible Hadamard matrices is constructed as described in chapter 5 and a list of Golay pairs can be used as described in chapters 4 and 7. Section 7.2 indicates that with a normalized correlation thresholds of 0.25 the list of Golay pairs should have size 1. However, the use of a larger list does make the task of the observer harder and should be considered advantageous. The approach does ensure that the correlation between generations is below the specified threshold. Thus one generation of the spreading code cannot be used to predict another for the purposes of jamming or eavesdropping. It should be noted that the recency list approach for the internal padding vectors does of itself ensure that the correlation between close generations is below the specified threshold. This allows more than one Golay pair to be used, as it seems relatively unimportant if the correlation between very distant generations is a little above the specified threshold. The main role of the Hadamard matrix evolution then becomes important from the second viewpoint, rather than from both viewpoints.

The results of chapter 10 indicate that complete theoretical security is not possible. The initial assumption that LS codes are less secure than codes in

use today (such as maximal length sequences or Gold codes) proves to be false. As a result of the Berlekamp-Massey algorithm, any code that can generated by a linear feedback shift register can theoretically be determined very early in the transmission of a single bit. It is shown in chapter 10 that LS codes cannot theoretically be determined as early in the transmission of a bit as a LFSR code, but nevertheless are still determined relatively early. Thus they could, for example, jam a large proportion of the bit transmitted, and therefore prevent correct reception of the bit. This theoretical consideration ignores the time necessary to carry out the necessary computations. Thus it is necessary to consider the time required for computations, both for the user and for the observer. Theoretically it seems that the code should be evolved for every bit to be transmitted. This may already create a challenging computation for the user transmitting the sequence of bits. The challenge for the observer in carrying out the necessary computations is considerably greater. Indeed it does appear the computation could hardly be completed in the period for which its result might be useful. Thus it might be possible to demonstrate the security of the system by considering the relative latencies for the user and the observer. However, as the computations are likely to require special purpose hardware or firmware, such considerations are beyond the scope of this thesis.

From the second viewpoint the true weakness of LS codes becomes apparent. All components of the construction are shared by all users except for the actual Hadamard matrix row of the user. This problem can be solved if a Gold or Gold-like code is used to construct the Hadamard matrix. However, even with this solution, it does not seem plausible that an algorithm could be supplied to each user for generating the individual sequence of Hadamard matrix rows, without giving the user the ability to generate the corresponding Hadamard matrix rows of other users. The consequence of this appears to be that the system can only be made secure if sequences of Hadamard matrix rows (or sets of Hadamard matrix rows for selection by all users with the common random number generator) are computed centrally, as described in mechanism 7 of chapter 8. The sequence or set for each individual user would then have to be circulated as a private key to the transmitter and all its receivers. However, repetition could not be allowed, so the key would have to be enormous. Only with the use of such enormous keys (and the solution of the consequent key distribution issues) could the system be considered truly secure. It would be secure with respect to this second viewpoint because the observer might know everything about the common random number generator, the evolving sequences of Golay pairs, the evolving sequences of internal padding vectors and one sequence of Hadamard matrix rows, but would have no way of determining the sequence of Hadamard matrix rows of other users.

Two important issues remain for further study. The first concerns the la-

tencies for both the user and the observer. Once the hardware and firmware to be used is determined, a study of the latencies is necessary to determine how quickly evolution can take place. This study will also determine whether the system is actually secure, or whether the theoretical ability of the observer to determine the spreading codeword early in the transmission of a bit is relevant. The second issue is the removal of the restriction that the length of the Golay pairs and the order of the Hadamard matrices be powers of 2. Permitting Golay pairs of lengths that are multiples of 10 and 26 gives a little more flexibility in designing the codeword length. Allowing other orders for Hadamard matrices again gives more flexibility, but creates major challenges in replicating the work of chapter 5.

Bibliography

[1] E.R. Berlekamp, "Algebraic coding theory", *McGraw-Hill*, New York, ch. 7, 1968.

[2] P.B. Borwein and R.A. Ferguson, "A complete description of Golay pairs for lengths up to 100", *Mathematics of Computation*, vol. 73, no. 246, pp. 967-985, 2004.

[3] R. Carraghan and P. Pardalos, "An exact algorithm for the maximum clique problem", *Operations Research Letters*, vol. 9, pp. 375-382, 1990.

[4] R. Craigen, Chapter 24 of "The CRC handbook of combinatorial designs" (ed. C.J. Colbourn and J.H. Dinitz), Boca Raton, Florida, CRC Press, 1996.

[5] J.A. Davies and J. Jedwab, "Peak-to-mean power control in ODFM, Golay complementary sequences, and Reed-Muller codes," *IEEE Trans. Inform. Theory*, vol. 45, no. 7, pp. 2397-2417, November 1999.

[6] R. De Gaudenzi, C. Elia and R. Viola, "Bandlimited quasisynchronous CDMA: A novel satellite access technique for mobile and personal communication systems," *IEEE J. Select. Areas Commun.*, vol. 10, pp. 328-343, February 1992.

[7] E.H. Dinan and B. Jabbari, "Spreading codes for direct sequence CDMA and wideband CDMA cellular networks," *IEEE Communications Magazine*, pp. 48-54, September 1998.

[8] S. Eliahou, M. Kervaire and B. Saffari, "A new restriction on the lengths of Golay complementary sequences," *J. Combinatorial Theory (A)*, vol. 55, pp. 49-59, 1990.

[9] M.J.E. Golay, "Complementary series," *IRE Trans. Inform. Theory*, vol. IT-7, pp. 82-87, April 1961.

[10] M.J.E. Golay, "Multislit spectroscopy," *J. Opt. Soc. Amer.*, vol. 39, pp. 437-444, 1949.

[11] J. Hadamard, "Resolution d'une question relative aux determinants," *Bull. Sci. Math.*, vol. 2, pp. 240-246, 1893.

[12] H. Kharaghani and B. Tayfeh-Rezaie, "A Hadamard matrix of order 428," *J. Combinatorial Designs*, vol. 13, pp. 435-440, 2005.

[13] D. Li, "A high spectrum efficient multiple access code," *Chinese Journal of Electronics*, vol. 8, pp. 221-226, July 1999.

[14] F. J. MacWilliams and N. J. A. Sloane, "The theory of error-correcting codes," Amsterdam, The Netherlands, North-Holland, 1977.

[15] J.L. Massey, "Shift-register synthesis and BCH decoding", *IEEE Trans. Inform. Theory*, vol. IT-15, no. 1, pp. 122-127, 1969.

[16] R. Paley, "On orthogonal matrices," *Journal of Mathematics and Physics*, vol. 12, pp. 311-320, 1933.

[17] S.O. Sanusi, D.H. Smith, R.A. Jones and S. Perkins, "The application of frequency assignment techniques in spreading code assignment," submitted.

[18] D.V. Sarwate and M.B. Pursley, "Crosscorrelation properties of pseudorandom sequences," *Proc. IEEE*, vol. 68, no. 5, pp. 593-619, May 1980.

[19] B. Sklar, "Digital communications: fundamentals and applications," Second Edition, New Jersey, Prentice-Hall PTR, 2001.

[20] D.H. Smith, R.P. Ward and S. Perkins, "Gold codes, Hadamard partitions and the security of CDMA systems," submitted.

[21] S. Stańczak, H. Boche and M. Haardt, "Are LAS-codes a miracle?," *IEEE Global Communications Conference(GLOBECOM)*, San Antonio, TX., U.S.A., pp. 589-593, November 25-29 2001.

[22] J.J. Sylvester, "Thoughts on inverse orthogonal matrices, simultaneous sign-succession, and tessellated pavements in two or more colours, with applications to Newton's rule, ornamental tile-work, and theory of numbers," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 34, pp. 461-475, December 1867.

[23] X.H. Tang and P.Z. Fan, "Bounds on aperiodic and odd correlations of spreading sequences with low or zero correlation zone", *Electronics Letters*, vol. 37, no. 19, pp. 1201-1202, September 2001.

[24] X.H. Tang, P.Z. Fan and S. Matsufuji, "Lower bounds on correlation of spreading sequence set with low or zero correlation zone", *Electronic Letters*, vol. 36, no. 6, pp.551-552, 2000.

[25] X. Tang and W.H. Mow, "Design of spreading codes for quasi-synchronous CDMA with intercell interference", *IEEE J. Selected Areas in Comm.*, vol. 24, no. 1, pp. 84-93, 2006.

[26] C.C. Tseng and C. L. Liu, "Complementary sets of sequences", *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 644-652, September 1972.

[27] L.R. Welch, "Lower bounds on the maximum cross correlation of signals", *IEEE Trans. Inform. Theory*, vol. IT-20, no. 3, pp. 397-399, May 1974.

[28] J. Williamson, "Hadamard's determinants theorem and the sum of four squares", *Duke Math. J.*, vol. 11, pp. 65-81, 1944.

[29] K. Yang, Y. Kim and P. V. Kumar, "Quasi-orthogonal sequences for code division multiple access systems", *IEEE Trans. Inform. Theory*, vol. 46, no. 3, pp. 982-993, May 2000.